



Universiteit  
Leiden

# Master Computer Science

StratoNet, a deep learning approach to high content screening feature extraction

Name: Michel van Elswijk  
Student ID: S2884852  
Date: 25/07/2024  
Specialisation: Bioinformatics  
1st supervisor: Dr. E.M. Bakker  
2nd supervisor: Prof. Dr. M.S.K. Lew  
Daily supervisor: Dr. ing. W.A. Omta

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)  
Leiden University  
Niels Bohrweg 1  
2333 CA Leiden  
The Netherlands

# Acknowledgements

Before proceeding, I would like to express my profound gratitude to all those who have supported me while completing this Master's thesis.

First and foremost, I extend my sincerest thanks to my daily supervisor, Dr. ing. W.A. Omta, for his guidance, feedback, and encouragement throughout this project. Their expertise and dedication have been crucial in shaping the direction and quality of this work.

In addition, I express my gratitude to Dr. E.M. Bakker for their insightful recommendations and constructive feedback, which have significantly elevated the rigor and depth of this work.

I want to extend my gratitude to my colleagues and friends at Core Life Analytics, whose camaraderie and intellectual exchange have provided a stimulating environment for academic growth. In particular, I would like to acknowledge Drs. J Mooij for their technical assistance and collaborative spirit.

I would like to thank my family for their unconditional love and support. To my parents, who have always believed in me and encouraged my educational pursuits, and to my partner, Emma, for their patience, understanding, and constant motivation for my endeavors.

I would like to express my gratitude to all those who have provided me with unwavering support and encouragement throughout the past year.

**Keywords:** Python, Deep learning, Neural Network, Feature Extraction, High Content Screening, StratoNet

### **Abstract**

This thesis presents the development of a deep learning feature extraction network tailored explicitly to the High Content Screening (HCS) domain. The proposed StratoNet combines existing neural network architectures, including ResNet, ECAnet, and GapNet. It integrates efficient channel attention to focus on objects and quantify features that represent them. It incorporates intermediate feature extractions for low, medium, and high-level cell features. StratoNet aims to extract the most beneficial features from the latent space of a convolutional neural network, which can correctly identify and differentiate between the phenotypical changes a drug or compound produces on a cell, also called Mechanisms of Action (MoA). The experimental results demonstrate that StratoNet can detect phenotypical changes and differentiate between different mechanisms of action in high-content screening data. StratoNet exhibited comparable performance to CellProfiler, although CellProfiler demonstrated better performance on classes with fewer phenotypic changes.

# Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Introduction</b>   | <b>1</b>  |
| <b>2</b> | <b>Related research</b>                                       | <b>3</b>  |
| 2.1      | Feature Extraction in HCS . . . . .                           | 3         |
| 2.2      | JUMP-Cell Painting Consortium . . . . .                       | 4         |
| 2.3      | Channel Attention For Convolutional Neural Networks . . . . . | 5         |
| <b>3</b> | <b>Fundamentals of HCS</b>                                    | <b>5</b>  |
| <b>4</b> | <b>Methodology</b>  | <b>10</b> |
| 4.1      | Data Description . . . . .                                    | 10        |
| 4.1.1    | ImageNet . . . . .  | 10        |
| 4.1.2    | JumpCP . . . . .  | 11        |
| 4.1.3    | Caie . . . . .  | 12        |
| 4.2      | StratoNet model . . . . .                                     | 12        |
| 4.3      | Channel Specific Feature Extraction . . . . .                 | 14        |
| <b>5</b> | <b>Experiments</b>  | <b>15</b> |
| 5.1      | Prerequisites . . . . .                                       | 15        |
| 5.2      | StratoNet training . . . . .                                  | 15        |
| 5.3      | Feature extraction evaluation . . . . .                       | 16        |
| <b>6</b> | <b>Results</b>  | <b>17</b> |
| 6.1      | StratoNet Classification Performance . . . . .                | 17        |
| 6.2      | High Content Screening Feature Extraction . . . . .           | 17        |
| 6.2.1    | Classification Methods . . . . .                              | 18        |
| 6.2.2    | Filtered Classification Methods . . . . .                     | 19        |
| 6.2.3    | Transferability of StratoNet . . . . .                        | 21        |
| 6.2.4    | StratoNet Features vs CellProfiler Features . . . . .         | 22        |
| 6.2.5    | Feature Mappings . . . . .                                    | 23        |
| <b>7</b> | <b>Discussion</b>   | <b>24</b> |
| <b>8</b> | <b>Conclusion</b>   | <b>25</b> |
| 8.1      | Future work . . . . .   | 25        |
|          | <b>References</b>   | <b>26</b> |
|          | <b>Appendix</b>   | <b>29</b> |

# Abbreviations

|   |    |
|---|----|
| <b>AI</b> Artificial Intelligence . . . . .                         | 1  |
| <b>CNN</b> Convolutional Neural Network . . . . .                   | 1  |
| <b>DL</b> Deep Learning . . . . .                                   | 1  |
| <b>MoA</b> Mechanism of Action . . . . .                            | 1  |
| <b>UMAP</b> Uniform Manifold Approximation and Projection . . . . . | 16 |
| <b>HCS</b> High Content Screening . . . . .                         | 1  |
| <b>GAP</b> Global Average Pooling . . . . .                         | 2  |
| <b>AWS</b> Amazon Web Services . . . . .                            | 15 |
| <b>MRQ</b> Main Research Question . . . . .                         | 2  |
| <b>SQ</b> Sub Research Question . . . . .                           | 2  |
| <b>Jump-CP</b> JUMP-Cell Painting . . . . .                         | 4  |
| <b>U2OS</b> Human Osteosarcoma Cell Line . . . . .                  | 4  |
| <b>ReLu</b> Rectified Linear Unit . . . . .                         | 7  |
| <b>ECA</b> Efficient Channel Attention . . . . .                    | 2  |
| <b>ECANet</b> Efficient Channel Attention Network . . . . .         | 2  |
| <b>ResNet</b> Residual Neural Network . . . . .                     | 2  |

# 1 Introduction

In the current era, characterized by the fast and unstructured flow of data, deep learning methods help to refine this data into powerful insights across numerous domains. The domain of High Content Screening (HCS) plays an essential role in selecting promising chemical compounds, a critical step in the costly process of drug development, which is estimated to cost between 314 million and 2.8 billion [1] for an individual drug. Selecting chemical compounds with the highest probability of clinical success from numerous chemical compounds is a complex but essential task in bringing new promising drugs to market. The use of Artificial Intelligence (AI) and particularly Deep Learning (DL) has attracted considerable attention in the field of HCS and has sparked debate in a wide range of other fields [2]. DL has started successfully providing valuable characteristic insights into the essential features of promising drug candidates before more money is invested.

The Mechanism of Action (MoA) is an essential characteristic for drug candidates. It defines how a drug interacts with cells and the effects it generates. Researching the MoA for different compounds is crucial to understanding their biological reactions and potential side effects. Clinical trials require drug candidates to undergo MoA screenings. Determining the MoA for various chemical compounds is labor-intensive and challenging. For example, some compounds may induce more than one MoA (i.e., Polypharmacology). The presence of one compound may influence the interaction of more than one target, thereby complicating the classification process [3].

The HCS field has shown increasing interest in conducting experimental studies using DL. Several studies examine the advantages and limitations of DL in HCS [4]. The primary focus of DL in HCS centers on addressing classification problems, specifically the categorization of a fixed set of chemical compounds into various MoA classes [5, 6].

The domain of HCS has become more interested in experimental studies with DL. With studies discussing the benefits and limitations of DL in HCS [4]. The focus of DL in HCS tends to be on solving a classification problem with a fixed set of MoA classes [5, 6]. This constrains exploratory research by restricting a DL experiment to domain-specific pre-existing classes and known phenotypes. Recent approaches to using Convolutional Neural Network (CNN) methods for MoA classification have proven to be a success [7]. The backbones of these implementations consist of pre-trained neural networks. However, these neural networks are not explicitly designed for high-content screening and, although seemingly successful, may not capture the true potential of these techniques when used for feature extraction. Despite this, the recent successes of CNN methods in MoA classification demonstrate their effectiveness in the field.

Traditionally HCS has utilized software such as CellProfiler [8], IN Carta [9] or HCS Studio [10] to segment and quantify data. These approaches are based on different segmentation methods from which the user can choose (e.g., global thresholding, subtraction, watershed). These segmentation methods enable identifying objects and their differentiation from background noise. A standard set of measurements is applied to the identified objects to quantify their objects. The primary limitation of this approach is that the image analysis software employs a protocol for each experiment, which can vary from one experiment to the next. The segmentation methods and other image preprocessing, such as illumination corrections, are gathered and performed sequentially in a CellProfiler protocol format. These protocols can be modified by their end users, allowing them to optimize them for the specific datasets. Such protocol defines how these objects are identified and whether they are recognized as valid objects. The end users can tweak the protocol by adjusting the parameters to the best settings for their dataset to optimize it for what the end user considers to be the highest quality data output. This can result in a set of features that are inconsistent, incompatible, and irreproducible, leading to highly subjective results. These features can vary in meaning and pattern from experiment to experiment. Therefore, besides image acquisition, object segmentation has to be considered one of the most fundamental processes in HCS [11].

This work contributes to the field of HCS screening by developing a deep learning convolutional neural network, the StratoNet, that employs efficient channel attention and intermediate feature extractions. Combining these techniques will facilitate the creation of a standardized set of reproducible features consistent across experiments. In addition, StratoNet will limit missing data in HCS, which is commonly caused by segmentation errors. The end user must be limited to modifying or influencing this process in any way in order to ensure consistency and reproducibility.

This thesis addresses several key research questions, which help divide the problem into four independent questions. These questions are divided into Main Research Question (MRQ) and Sub Research Question (SQ) to provide a structured approach to this work.

### Main research question

**MRQ** Can we define effective methods for high-content screening that produce reproducible features?

### Sub research questions

**SQ1** Is it feasible to establish a standardized set of reproducible and effective features extracted through a convolutional neural network for high-content screening?

**SQ2** What strategies can be employed to utilize convolutional neural networks for extracting reproducible and effective features in high-content screening across various high-content screening datasets?

**SQ3** How do deep learning features compare to established gold standard image processing features in facilitating class separation for predicting the Mechanism of Action in high-content screening?

This thesis proposes a novel combination of well-established CNNs and channel-specific intermediate feature extractions as a potential solution for generating effective and reproducible features in HCS. Combining the Efficient Channel Attention (ECA) module and intermediate feature extractions will serve as the basis for this novel combination, named StratoNet. StratoNet, amongst other CNNs, will extract a standardized feature set from multiple datasets, thereby enabling the comparison of multiple datasets. For this thesis, Residual Neural Network (ResNet) and Efficient Channel Attention Network (ECANet) will be used to show the performance and set the baseline for neural networks. Eventually, StratoNet will be demonstrated as the novel solution and compared to the previous models. The model's transferability across different experiments and datasets will be tested. Subsequently, StratoNet will be trained and tested compared to the industry standard, CellProfiler, in the HCS field. The entire pipeline will be accessible through a Software-as-a-Service application within the StratoVerse, enabling users to generate a comprehensive feature set for their experiment.

The rest of this thesis is organized as follows: Section 2 reviews the current state of DL in HCS and highlights how our methods differ from previous work. Highlighted is related research on CNN attention variants and Global Average Pooling (GAP) intermediate layer extractions. Section 3 provides a brief overview of the foundational concepts relevant to our study. Section 4 introduces the datasets used, details the modifications made to the CNNs, and presents the proposed StratoNet model for HCS. The experimental setup is described in Section 5. The results are analyzed in Section 6 and discussed in Section 7. Finally, Section 8 outlines the research questions and suggests directions for future work.

This thesis contributes to the innovative concept of extracting multiple features from various positions that are phenotypically significant for HCS, incorporating this into a novel neural network called StratoNet.

This master's thesis is a collaborative effort between Core Life Analytics and Leiden University. This study proposes a novel and reproducible methodology for feature extraction in HCS, adhering to

traditional academic conventions in citation, footnotes, and formatting. It addresses critical issues and suggests improvements for HCS feature extraction. This work includes a comprehensive analysis from academic and practical perspectives, leveraging the expertise of both organizations. Through this partnership, we aim to enhance understanding in the field and provide viable solutions and potential benefits to the industry.

## 2 Related research

This Section discusses work and approaches related to feature extraction in HCS. Furthermore, relevant work on CNNs and attention layers will be examined. Finally, the combination of both approaches will be discussed, considering the significance of the earlier related research, and the state of the art will be established.

### 2.1 Feature Extraction in HCS

The 2021 update to CellProfiler 4 improves speed and usability and introduces additional features to the segmentation-based HCS image quantifier [12]. CellProfiler is a free, open-source image quantification tool that allows researchers to develop pipelines for processing HCS images into interpretable measurements. It is currently considered the gold standard in the field of HCS [13]. Typical CellProfiler pipelines involve scaling images between the  $[0, 1]$  range followed by a background illumination correction before measurement. CellProfiler can then segment and use its pipeline with a set of measurements. Starting with CellProfiler version 3, pre-trained neural networks can classify, e.g., background, nucleus interior, or nucleus border with the ClassifyPixels-Unet segmentation module [8]. Other methods that involve the extraction of raw DL-based features from HCS images are not yet available. The focus of this work is mainly on the command-line version of CellProfiler. Our work aims to enable DL feature extraction for HCS and demonstrate its relevance and strength for the field.

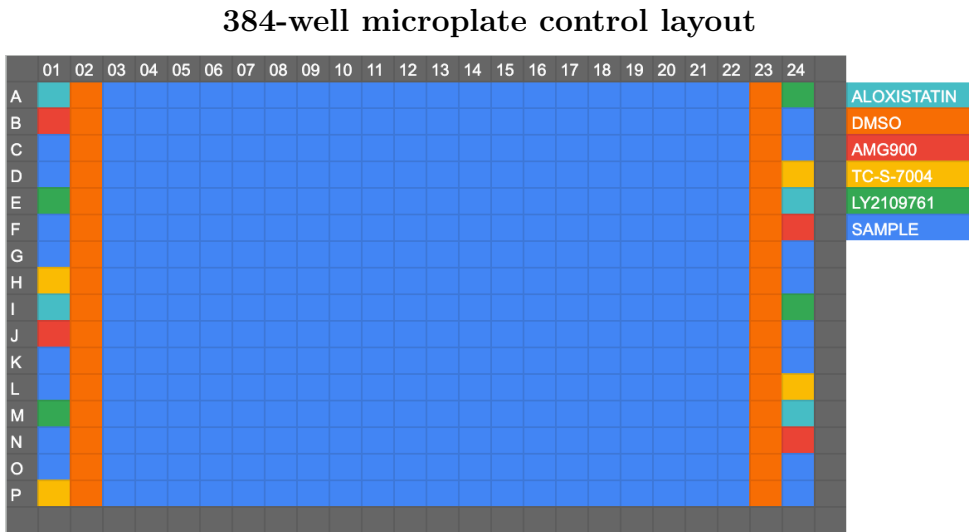
As previously mentioned, past research on feature extraction in HCS has focused on a classification problem with a predetermined set of MoA classes. *Wong et al.*'s work (2023) [14] centers on the analysis of latent representations of HCS data, aiming to minimize the distance between embeddings with the same MoA class and maximize the distance between different MoA classes in the latent space. This approach groups identical MoA classes while separating non-identical ones. Our work aims to address the same problem differently. Rather than interpreting images as a stack of channels, this work will extract channel-specific features. Extracting channel-specific features addresses the issue when a particular channel (e.g., DNA, RNA) is missing. By filtering out the features of the missing channel from the dataset, comparisons can still be made between datasets that do not precisely match channel-wise. Furthermore, this work will utilize a more general approach using a pre-trained neural network combined with a more novel channel attention CNN.



## 2.2 JUMP-Cell Painting Consortium

Sufficient qualitative data is the most critical resource for a successful DL experiment. With an adequate amount of data, a deep learning application will be able to learn the necessary patterns within a dataset or will become focused on a narrow set of known classes. This issue represents a significant challenge for numerous domains that have thus far been unable to utilize DL. [15]. A significant HCS dataset was announced on July 10, 2020, and subsequently released by the Broad Institute on March 24, 2023 [16]. This dataset, hereafter referred to as JUMP-Cell Painting (Jump-CP), is derived from cellular imaging and is designed to create the most comprehensive publicly available Cell Painting dataset. It aims to recommend optimal methodologies for conducting Cell Painting experiments while developing community-driven data analysis and storage techniques. In total, there are 116,750 unique compounds screened on the Human Osteosarcoma Cell Line (U2OS) within this dataset. Furthermore, the dataset includes additional assays, such as gene overexpression and gene knockout using CRISPR-Cas9. However, these will not be discussed in this work. The dataset is approximately 116 terabytes, encompassing 1.6 billion cells and their respective single-cell profiles. The dataset was acquired from 12 different sources in five replicates using different instruments and microscopes, increasing its robustness as a reference dataset. Throughout the sources, different plate layout designs are used. It is important to note that the controls vary depending on the specific plate layout utilized for a given assay. This work focused on Source 8 data; the plate layout for these assays is shown in Figure 1.

The publicly available dataset also provides quantified data using CellProfiler. This data is accompanied by the protocols used for CellProfiler. The version of CellProfiler utilized was v3, as referenced in Chandrasekaran *et al.* [16]. This study will evaluate the effectiveness of deep learning feature extraction techniques in HCS and compare the results with the quantified data from Source 8. Additionally, the protocols will be re-evaluated to determine if similar results are achieved when considering the MoA predictions.



**Table 1:** The plate layout for source 8 from JumpCP was utilized throughout this work. Most of this plate layout comprises different compounds located in the sample wells (blue). There are five control compounds: Aloxostatin (light blue), DMSO (orange), AMG900 (red), LY2109761 (green), and TCS7004 (yellow).

## 2.3 Channel Attention For Convolutional Neural Networks

In recent years, the field of DL has undergone a significant shift, with the heavy adaptation of attention mechanisms. Attention mechanisms have become universal, with almost everyone today employing some form of attention mechanism when utilizing OpenAI's chatGPT or Google's Bard [17, 18]. Attention mechanisms have gained immense popularity and have been extensively adapted in numerous applications [19]. The essence of attention in computer vision models is to focus on specific sensory input while discarding nonessential information such as background. This work builds upon the concept of attention, utilizing the work of Wang et al. 2020 [20] as a foundation to develop an attention-based feature extraction model specifically for the field of HCS. In their study, Wang et al. 2020 [20] developed an ECA module that employs a squeeze-and-excitation block to reduce the complexity of the model without resorting to dimensionality reduction. This module was proposed as part of the ECA-Net CNN architecture. This work proposes utilizing the Efficient Channel Attention ECA module within a ResNet architecture for feature extraction. It draws inspiration from Hofmarcher et al. 2019 [21] research on the use of CNNs for predicting biological assays. Their study highlighted CNNs as superior to traditional, expensive cell segmentation-based methods for feature extraction. The GAP intermediate feature extraction layers, particularly noted for their effectiveness in image localization, significantly influence this project [22, 23]. Additionally, the GAP layers are the foundation for the ECA module, which will be integrated into this study.

This thesis aims to provide the StratoNet model, which enables effective low, medium, and high-level feature extraction for the HCS domain. The goal is to extract the most beneficial features from the CNNs. To date, almost all HCS DL implementations have focused on correctly classifying a fixed set of MoA classes. This work aims to fill the gap and provide the StratoNet capable of correctly finding and separating MoA classes based on the latent space features of the CNN.

Many efforts have been made within HCS to leverage the potential of DL in the context of drug research. However, this work demonstrates that integrating domain-specific knowledge from HCS into DL methods could enhance the field's effectiveness and impact.

## 3 Fundamentals of HCS

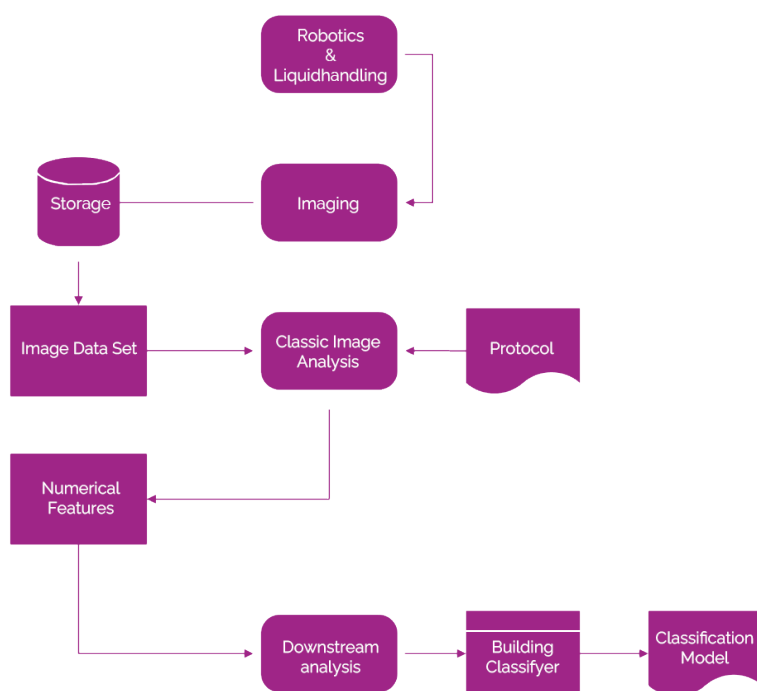
A comprehensive grasp of the fundamental principles is essential in the HCS field. This Section attempts to provide a comprehensive overview of the field, discussing concepts such as cell painting, MoA, CellProfiler, and the software that combines everything: StratoVerse, developed by Core Life Analytics [24]. These elements serve as the foundation for HCS and practical applications, guiding research and practice within the field.

### *Mechanism of Action*

The term MoA refers to the biochemical processes by which a drug produces its effect. These processes typically involve the specific molecular targets to which the drug binds, such as enzymes or receptors. These interactions can lead to observable effects, such as changes in cell function or protein activity, which are quantified using feature extraction models. Receptor sites have specific affinities for drugs based on their chemical structures, leading to therapeutic effects even when drugs do not bind directly to these receptors. The knowledge of the MoA of a drug may assist in assessing its safety and identifying its effects on the body. It may also facilitate the determination of the optimal dose of a drug and identify patients who are most likely to respond to treatment.

### HCS Pipeline

**HCS** uses automated microscopy to image cells exposed to different chemical libraries, antibodies, or bioactive reagents. Automated image analysis generates a multi-parametric numerical profile to characterize cellular changes, making **HCS** critical in drug discovery programs. Methods such as CellPainting extract thousands of features (e.g., area, shape, size, count, texture, intensity) from images, creating detailed profiles for unsupervised or supervised data analysis. The general procedure of a typical **HCS** experiment encompasses a series of stages as illustrated in Figure 1. The initial phase encompasses the comprehensive liquid handling and imaging process. Liquid handling involves the preparation of microplates, which entails the seeding of cells, the addition of solvents, reagents, and fluorescent chemicals, and subsequent imaging. The resulting image data can be conceptualized as the "raw data". The images are stored in a database and retrieved once the assay is prepared for quantification. Subsequently, the images undergo a pipeline that quantifies them to numerical values. Following this, the numerical data is prepared for processing and analysis.



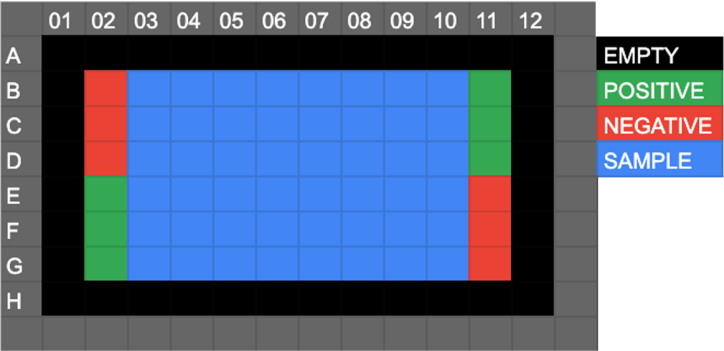
**Figure 1:** A simplified workflow diagram for traditional HCS. It begins with robotics and liquid handling and ends with the classification model

### Cellpainting

Cellpainting is an **HCS** screening technique used in biological research and drug discovery. Cells are stained with a combination of fluorescent dyes that target different cellular components. Image processing and machine learning algorithms extract features and measurements from the images, including cell shape, size, texture, and subcellular localization. Cell painting provides a holistic view of cellular responses, allowing researchers to study cellular phenotypes and identify the effects of different treatments or conditions.

In **HCS**, imaging work is conducted using microplate formats such as the 96-, 384-, and 1536-well plates, as illustrated in Table 2. The plate layout designates the locations of controls and samples on the microplate.

Example 96-well microplate



**Table 2:** The example 96-well microplate includes positive controls (green), negative controls (red), and samples (blue). Each well is uniquely identified by its type, column, and row.

*CellProfiler*

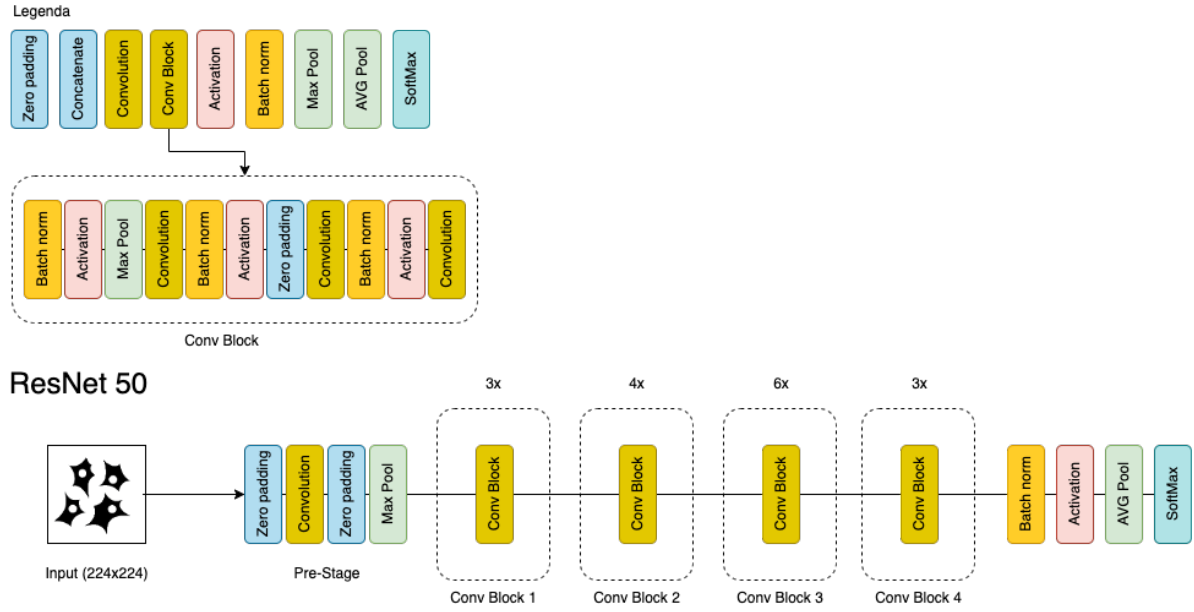
CellProfiler is an image quantification tool designed to enable biologists without experience in computer vision or programming. It is an open-source, free software program. Advanced algorithms are included for image analyses. A protocol must be provided to work with CellProfiler. Protocols are modules of advanced algorithms combined sequentially to form a pipeline. These protocols can identify and measure biological objects and features [25].

ResNet and ECANet can be utilized to replace the image quantification step for HCS. Illustrative applications of CNNs encompass the recognition of pathways and phenotypes. These techniques facilitate the resolution of complex image analysis problems encountered by biologists. The ResNet architecture represents a prominent example of a CNN. There exist diverse depths of DL architectures.

*ResNet*

The ResNet50 CNN illustrated in Figure 2 comprises 50 layers and employs residual blocks to facilitate the training of deep networks. The fundamental concept of ResNet is the residual connections to the network, which addresses the vanishing gradients problem, which was a significant challenge for deeper neural networks. Residual connections represent a shortcut connection between residual blocks that circumvent one or more layers. These connections are also referred to as the identity. The residual connections serve to maintain the information flow throughout the network. Other essential elements of the architectural framework include convolutional layers, batch normalization, and Rectified Linear Unit (ReLU) activations. The legend offers a comprehensive overview of components and their respective roles within the network.

## ResNet Architecture

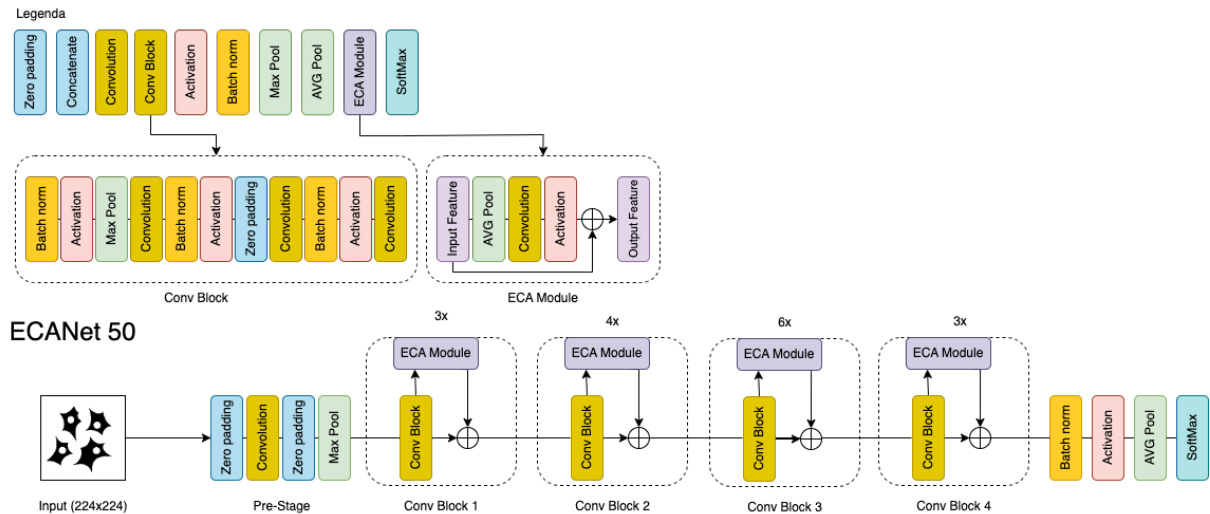


**Figure 2:** Schematic representation of the ResNet50 architecture

### ECANet

A more recent and advanced CNN network is the ECANet. ECANet employs ECA modules, which are attention models [20]. The ECANet CNN, illustrated in Figure 3, incorporates the ECA mechanism into the existing ResNet50 CNN model. This enhances the representation power of the network by adaptively adjusting the weights of different channels without significantly increasing complexity. The key components of the architecture include convolutional layers, batch normalization, ReLU activations, and the ECA module, which computes channel attention efficiently through a local cross-channel interaction strategy. The legend provides a comprehensive overview of these components and their roles within the network.

## ECANet Architecture



**Figure 3:** Schematic representation of the ECANet50 architecture.

### *StratoVerse*

While not a prerequisite for this thesis, the StratoVerse was employed throughout the research. StratoVerse is an intuitive web-based application developed by Core Life Analytics, a University Medical Center Utrecht spin-off for pharmaceutical companies, biotech companies, and academic life science groups. The StratoVerse represents a combination of applications for HCS. Each application in the suite has its purpose. StratoStore stores the images and makes them available to any application within the suite. StratoViewer allows the images to be viewed in a structured manner for easy data access. It includes applications that quantify the images, such as CPUltra, a parallelized version of CellProfiler, and StratoFeatures, which is how this work is made available in the cloud environment. Additionally, the StratoMineR application allows for the analysis of all data uploaded to StratoVerse. In the cloud environment, CPU—and GPU-based machines are available for computation. It is possible that this work could have been completed without the use of StratoVerse. While StratoVerse did facilitate data management and access, handling vast amounts of data was straightforward and efficient.



**Figure 4:** The three principal applications of the StratoVerse suite

### *Performance measures*

The performance of the different models needs to be tested across all datasets. For the ImageNet-1k dataset, both top-1 and top-5 accuracy metrics are used. Top-1 accuracy measures how often the model's highest confidence prediction matches the actual label, while top-5 accuracy measures how often the correct label is among the model's top five predictions, providing a broader assessment of the model's ability to classify correctly.

Three distinct classification methods are utilized to assess the efficacy of the feature extraction.

1. **Support Vector Machine (SVM)**, a supervised learning algorithm that identifies the optimal hyperplane to separate data points of different classes in a high-dimensional space, maximizing the margin between the closest points of the classes (e.g., support vectors).
2. **Random Forest (RF)**, an ensemble learning method that builds multiple decision trees and aggregates their predictions to improve accuracy and reduce overfitting.
3. **Multi-Layer Perceptron (MLP)**, a basic neural network with an input layer, hidden layers, and an output layer, where each node is fully connected. MLPs use activation functions like ReLU to model complex relationships.

Each method has unique strengths and weaknesses and performs differently on each dataset. Using all three methods minimizes the risk of the dataset benefiting a specific classifier method and reduces bias in the analysis.

The macro average, shown in Formula 1, is computed for all classification methods, which evaluates the performance by independently calculating metrics such as precision for each class  $i$  and then averaging the results by the number of classes  $N$ . This approach treats all classes equally, providing a balanced performance measure, which is especially beneficial for imbalanced datasets.

$$\text{Macro Average} = \frac{1}{N} \sum_{i=1}^N \text{Metric}_i \quad (1)$$

## 4 Methodology

Developing a new method inspired by related work could improve the reproducibility of HCS experiments. This chapter will focus on the following key components: the data used for training and application, StratoNet, the proposed novel DL architecture for feature extraction, and integrating these components into the application, such as extracting channel-specific features.

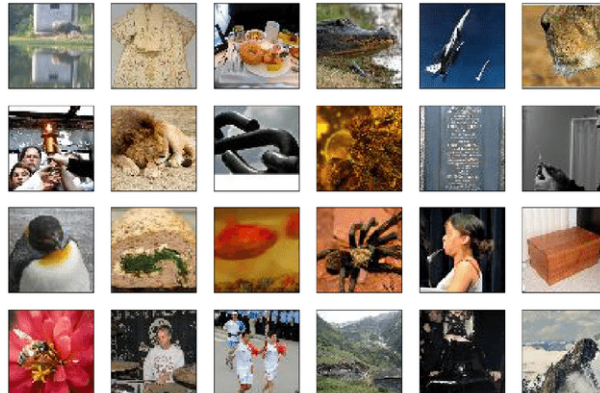
### 4.1 Data Description

This work employs the datasets ImageNet-1k, Jump-CP, and Caie et al., each serving a distinct purpose. These datasets can be divided into two primary categories. The first category encompasses datasets utilized for training neural network models, and the second category contains reference datasets utilized for applying the models and, thereby, feature extraction.

#### 4.1.1 ImageNet

The ImageNet dataset is one of the most well-known benchmark image datasets in deep learning. For the training steps, the ImageNet-1K is used [26] containing 159GB of data. The average resolution of the images is (469,387) in pixels and consists of 3 channels (e.g., RGB). This dataset is used on the models that are compared in this thesis and has been a part of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [27], an annual event held together with the ImageNet dataset to provide algorithms with the ability to classify images effectively.

Both the top-1 and top-5 error metrics are employed to assess the efficacy of the models, representing the percentage of cases in which the target label was not among the top-n predictions with the highest probability. Image classification methods that do not rely on CNN experience a performance plateau in the challenge, with numerous approaches unable to achieve an error rate below 25% for the top 5% guesses. In 2010, two professional annotators evaluated human performance on ImageNet, resulting in a top-5 error of 5.1% and 12% [26].



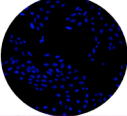
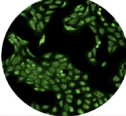
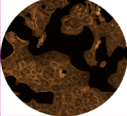
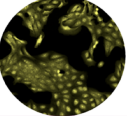
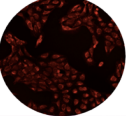
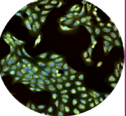
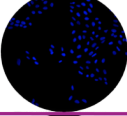
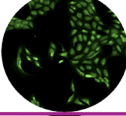
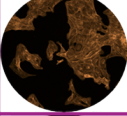
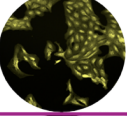

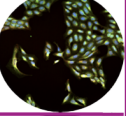
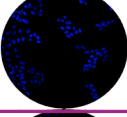
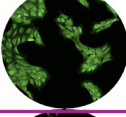
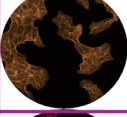
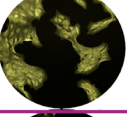
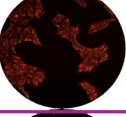
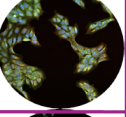
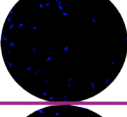
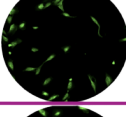
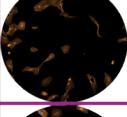
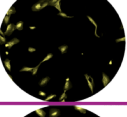
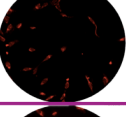
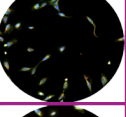

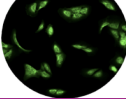
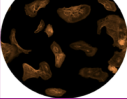
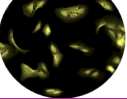
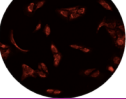
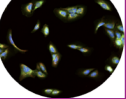
**Figure 5:** a random selection of images from the Imagenet-1k dataset



Figure 5 displays a random selection of images from the Imagenet-1k dataset, providing an overview of its contents. The preprocessing of the images followed the same practices as ResNet and ECA-net [28, 20]. In brief, two distinct preprocessing flows were employed. The preprocessing of the training set involved a resizing to dimensions of [256,256], followed by a random horizontal flip and a random [224,224] crop. Finally, the image is normalized between the range [0, 1] to be handled correctly by the neural network. The preprocessing of the test set consisted of a fixed resizing in [256,256]. However, there was always a center crop of [224,224] and no horizontal flip or other augmentations. The train and test split was identical to that employed in both ResNet and ECA-net [28, 20].

#### 4.1.2 JumpCP

Jump-CP is a dataset that has 116,750 unique compounds screened on the Human Osteosarcoma Cell Line U2OS [16]. The dataset is approximately 116 terabytes, encompassing 1.6 billion cells. The average resolution of the images is (1024,1024) in pixels and consists of 5 channels (e.g. MITO, GOLGI, RNA, ER, and DNA). The plate layout in Table 1 is used in source 8. This source has 216 plates, each of which is a 384-well microplate. Each well contains nine fields and five channels. Table 3 shows a sample of these images for the five channels and control compounds. All data, including the samples, is included. The illumination correction is not applied, and thus, it is omitted. The images are normalized between the range [0, 1] and resized to dimensions of [224, 224], as this is the input for the different neural networks. Each image is stacked to create a three-dimensional matrix to finalize the input for the neural network.

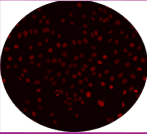
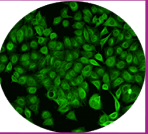
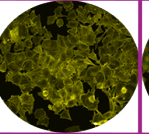
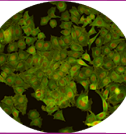
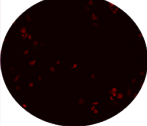
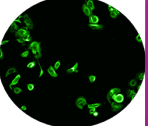
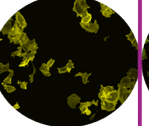
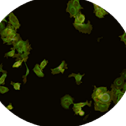
| Channel<br>Control | MITO  | GOLGI   | RNA   | ER  | DNA  | OVERLAY   |
|--------------------|---|---|---|---|--|---|
| DMSO               |  |  |  |  |  |  |
| Aloxistatin        |  |  |  |  |  |  |
| LY2109761          |  |  |  |  |  |  |
| TCS7004            |  |  |  |  |  |  |
| AMG900             |  |  |  |  |  |  |

**Table 3:** This Table shows a sample of images from the Jump-CP dataset. The first five columns show the fluorescent labels used, and the sixth column shows the cellular components overlay. The rows show different compounds added to the cells.



### 4.1.3 Caie

The work of *Caie et al.* [29] is employed as a control dataset for the feature extraction of the models. This dataset comprises 57 96-well microplates divided into two control classes. The average resolution of the images is (1280,1024) in pixels and consists of 3 channels (e.g. DNA, F-ACTIN,  $\beta$ -TUBULIN). Each well contains several fields and channels. No illumination correction has been included; thus, it has been omitted. Table 4 displays a sample of the images. The images have been normalized between the range [0, 1] and resized to dimensions of [224, 224]. Finally, each image is stacked to create a three-dimensional matrix, which serves as the final input for the neural network. The plate layout can be seen in Figure 2.

| Channel<br>Control | DNA   | F-ACTIN   | $\beta$ -TUBULIN   | OVERLAY   |
|--------------------|---|---|--|---|
| Negative           |  |  |  |  |
| Positive           |  |  |  |  |

**Table 4:** This Table shows a sample of images from the Caie dataset. The first three columns show the fluorescent labels used, and the fourth column shows the cellular components overlay. The rows show the different compounds added to the cells.

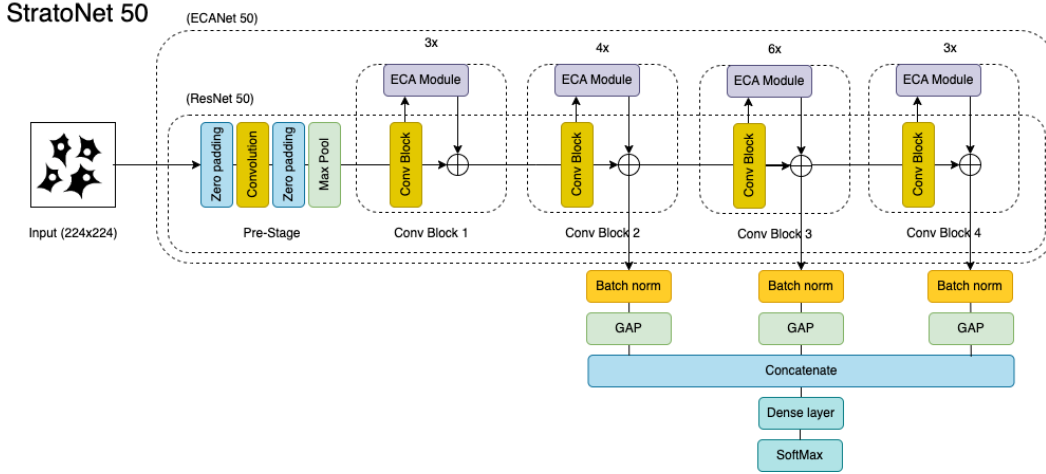
## 4.2 StratoNet model

The principal objective of this study is to develop a novel model that captures the essential characteristics of HCS data and, therefore, captures effective features. The architecture of the proposed StratoNet is depicted in Figure 6. This architecture contains 50 layers, although StratoNet is not constrained by this depth. The model can be dynamically modified, provided the underlying architecture remains intact. In Figure 6, the underlying components of both ResNet and ECANet can be seen separated by the dotted line. The last addition incorporates intermediate feature extractions, including GAP layers. The GAP layers help to reduce the spatial dimensions to a single value per feature map by computing the average of each feature map and, therefore, summarizing them.

The central concept is that cells exhibit both general and more detailed features that define their phenotype. By extracting intermediate features from various positions, it is believed that these positions will capture different levels of detail and, therefore, characterize the cell. Both the ECA Modules and the GAP layers help to focus on the foreground and, therefore, the objects in the images.

A script has been developed in Python 3.10.12 to construct the StratoNet dynamically, considering the hyperparameters [30]. TensorFlow was employed to construct and train neural networks. Appendix A provides a comprehensive list of all utilized packages. Additional requirements will be enumerated in the Section 5.

## StratoNet Architecture



**Figure 6:** Schematic representation of the StratoNet50 architecture.

In Figure 6, StratoNet-50 is displayed, however different depths of this architecture can be achieved by altering the number of residual blocks. Currently the setup displayed for StratoNet-50 consists of  $(1x, 3x, 4x, 6x, 3x)$  residual blocks sequentially. However, the architecture of StratoNet-18, StratoNet-34, StratoNet-101, and stratoNet-152 share an equivalent architecture with StratoNet-50 but have different numbers of residual blocks. The residual block counts are depicted in Table 5. It should be noted that from StratoNet-50, an additional convolution is added in conv2, conv3, conv4, and conv5.

|               | Conv1 | Conv2_x | Conv3_x | Conv4_x | Conv5_x |
|---------------|-------|---------|---------|---------|---------|
| StratoNet-18  | 1     | 2       | 2       | 2       | 2       |
| StratoNet-34  | 1     | 3       | 4       | 6       | 3       |
| StratoNet-50  | 1     | 3       | 4       | 6       | 3       |
| StratoNet-101 | 1     | 3       | 4       | 23      | 3       |
| StratoNet-152 | 1     | 3       | 8       | 36      | 3       |

**Table 5:** Residual block counts for different StratoNet architectures

Figure 6 illustrates the current state of StratoNet. However, hyperparameter optimization is still necessary. Therefore, the number of ECA modules, the depth of the classification layer, the learning rates, and other relevant hyperparameters will be tested in the Experiments Section 5. The training of StratoNet was based on a combination of ResNet and ECA-net [28, 20]. For each training round, a variant of the StratoNet model is selected and saved at each checkpoint if it demonstrates superior performance on the test set. A learning rate scheduler and a learning rate reducer prevent local optima and accelerate the learning process. The whole StratoNet training procedure is illustrated in Appendix B

### 4.3 Channel Specific Feature Extraction

After developing the StratoNet, this work focuses on extracting features using various methods. The first method to extract features is by using a CellProfiler pipeline. If a suitable protocol is provided for CellProfiler and images are processed through the pipeline, the object-based features are stored in a comma-separated file. The only additional step required is to merge the metadata with the results from CellProfiler for classification. We use the dataset that the Broad Institute provided with the **Jump-CP** data for consistency. The second part of this work focuses on extracting features using neural networks. For the neural network methods, the features are extracted from the latent spaces within the neural networks.

The feature extraction for the **CNN** methods is identical for each method, as they are all based on one another and progressively more complex. The primary distinction is that the feature extraction is derived from a single position in the case of **ResNet** and **ECANet**. In contrast, StratoNet has multiple intermediate feature extractions, as illustrated in Figure 6. The position of the feature extraction in the network has a considerable influence on the output features. Therefore, a preliminary experiment will be performed where the positive and negative controls for **Jump-CP** will be plotted against each other, ensuring that the distance between these two controls is as considerable as possible. The decided positions for feature extraction within the network will be the final positions for all neural networks. This is only possible because all neural networks are constructed upon one another. The channels (i.e., cellular stainings) were put through independently for each neural network model, so each channel has separate features instead of stacking the input as done in previous work. This was decided because stacking the input would make the model incompatible with other experiments if one channel failed or was missing. Running each channel through the network separately prevents this. Each feature extracted from the neural network will be given a naming convention that combines the channel ID and the numerical value for the extraction position from the neural network. A script was developed in Python 3.10.12 for the **CNN** models. This script gathered data from the cloud, combined the metadata, preprocessed it by the specifications outlined in Sections 4.1.2, 4.1.3, extracted features from the stated intermediate layers as seen in Algorithm 1, and saved the quantified datasets to the cloud environment of the StratoVerse.

---

**Algorithm 1** Intermediate Feature Extraction

---

**Require:** Trained neural network  $N$ , intermediate extraction layer(s)  $L$ , input images  $I$

**Ensure:** Intermediate features  $F$

```
1: Initialize an empty list  $F$ 
2: for Each input image  $i$  in  $I$  do
3:   for Each layer  $l_i$  in layers  $L$  do
4:     if Layer  $l_i$  exists in network  $N$  then
5:        $f_i \leftarrow L_i(X)$  ▷ Extract features from layer  $L_i$ 
6:       Append  $f_i$  to  $F$  ▷ Use naming convention combining feature position &
       image channel ID
7:     end if
8:   end for
9: end for
10: return  $F$ 
```

---

## 5 Experiments

To ensure the reproducibility of the experiments and the fairness of the comparisons in this work, this Section describes the software and hardware requirements, the datasets used, their preprocessing and augmentation steps, the training procedure for each model, and the evaluation of these models. In addition, the feature extraction pipeline is evaluated.

### 5.1 Prerequisites

All approaches in this work are executed within the same cloud-based environment of Amazon Web Services (**AWS**). While **AWS** was the chosen cloud provider, it is not the only option available. The code is executable on different cloud providers or even locally if the correct packages and hardware are provided. However, the image and numerical dataset storage are written using the **AWS** S3 API and must be used or replaced when using the feature extraction pipeline. For the hardware, two instance types are used from **AWS** mentioned in Table 6 [31].

| Usage description            | Instance Type | GPU | vCPU | Memory (GiB) | Instance Storage (GB) | Instance Price (\$) |
|------------------------------|---------------|-----|------|--------------|-----------------------|---------------------|
| Feature Extraction Pipeline  | x2gd.8xlarge  | NA  | 32   | 512          | 1900 NVMe SSD         | 3.20/h              |
| Model training & fine-tuning | p3.8xlarge    | 4   | 32   | 244          | 600 GP3               | 13.22/h             |

**Table 6:** Specifications of the different instance types used, prices are based on the eu-west-1 region on the 25th of April 2024

The initial instance, as detailed in Table 6, was utilized for feature extraction and was based on a CPU. As outlined in Table 6, the second instance was employed for training the StratoNet. This instance had four NVIDIA V100 Tensor Core GPUs, enabling parallel search for optimal hyperparameters. All coding was conducted using Python 3.10.12, and the specific packages utilized are detailed in Appendix Table 8.

### 5.2 StratoNet training

Any parameter that can be initialized before training the neural network model is considered a hyperparameter. These parameters include hyperparameters such as learning rate, decay rates, step size, and batch size. Additionally, more specific hyperparameters include the number of layers, the number of units in each layer, the dropout rate, and the type of activation functions such as ReLU, Sigmoid, and Tanh. For CNNs, there are also **CNN** specific hyperparameters related to the convolutional layer, such as window size, stride value, and pooling layers. During the hyperparameters optimization process, it was tried to stay as close as possible to the original architectures and parameters. However, in some cases, this work deviated from the original work. For example, the decision was made to change the original update rule from Stochastic Gradient Decent to the ADAM update rule. ADAM is robust if the hyperparameters are not optimal and will often converge fast enough, but Stochastic Gradient Decent can sometimes be faster if the correct hyperparameters are chosen. The train and test data were preprocessed as mentioned in Section 4.1.1. The train and test split was the same as in the original **ResNet** and **ECANet** papers. The train size was 1.281.167, and the test size was 50,000. The batch size was set to 128 because the GPU memory would overflow if set higher. This meant there were 10.009 iterations per epoch for a total of 100 epochs. Both top-1 and top-5 accuracy are used to evaluate the model's performance.

### 5.3 Feature extraction evaluation

After training the neural networks, a best-performing model is selected, and image data is fed through it to quantify the images. From both **Jump-CP** and *Caie et al.* [29] image data sets will be created. The data will be preprocessed as described in Sections 4.1.2 and 4.1.3, and the data will be fed through and extracted from the intermediate feature extractions decided on by the class separation based on the **Jump-CP** dataset. After each model quantifies the images, the numeric data is preprocessed and classified using three different classification methods. Before classification, the data is split into their respective train and test sets for the quantified data. As seen in the plate layout in Figure 1, there is a class imbalance on the plate. Stratified sampling addresses the class imbalance by taking 3000 points from the dataset. The Stratified sampling is executed after the train split to prevent duplicate samples and leakages within the classification. The result is the creation of a balanced training set. The test set should reflect the real-world distribution of classes. The real-world data is unbalanced; the test set should also be unbalanced to provide a realistic evaluation of the model's performance. However, to visualize the lack of performance for a specific class, a balanced experiment should also be conducted to rule out if a model lacks performance in specific cases. By utilizing three unique classification methods the risk of one model benefiting a specific classification method is limited. This allows the feature extraction methods to be tested instead of the classification methods. Finally, Uniform Manifold Approximation and Projection (**UMAP**) is displayed to illustrate the class separation to the reader. As well as the variance explained for the principal components to see how much variance is captured in the extracted features.

In the final experiment, a correlation plot analyzes the correlation between the CellProfiler features and the StratoFeatures generated from StratoNet. By visualizing this, it can be determined if there is any similarity between the features produced by the two feature extraction methods. Although computational hours are not the main point of interest, it is noteworthy to track execution time because these tasks tend to be time-intensive. Time is tracked for all feature extraction methods.

## 6 Results

This chapter presents the results of the StratoNet model and compares different feature extraction methods. It also includes detailed figures focusing on the characteristics of the StratoFeatures set created by StratoNet. Each section provides in-depth insight into how these elements could contribute to the overall goals of this work in the **HCS** data.

### 6.1 StratoNet Classification Performance

As previously stated, StratoNet is trained on ImageNet data for the initial experiment. The results are presented in Table 7. Additionally, the original accuracies for both ResNet50V2 [28] and ECA-net [20] are included in Table 7. The results show that StratoNet exhibits suboptimal performance in both Top-1 and Top5 accuracy on the ImageNet dataset. Additionally, it should be noted that StratoNet has a parameter count that is less than a third of that of other models. The reason for this is that the selected feature extractions occur relatively early in the network (see Appendix D). Therefore, the network's end is cut off because it is not used for feature extraction, which causes the amount of parameters to shrink.

| Model           | Accuracy (%) |       | Params<br>(millions) |
|-----------------|--------------|-------|----------------------|
|                 | Top-1        | Top-5 |                      |
| ResNet50V2 [28] | 75.20        | 92.52 | 25.6                 |
| ECANet50 [20]   | 77.48        | 93.68 | 29.5                 |
| StratoNet50     | 56.61        | 79.92 | 8.5                  |

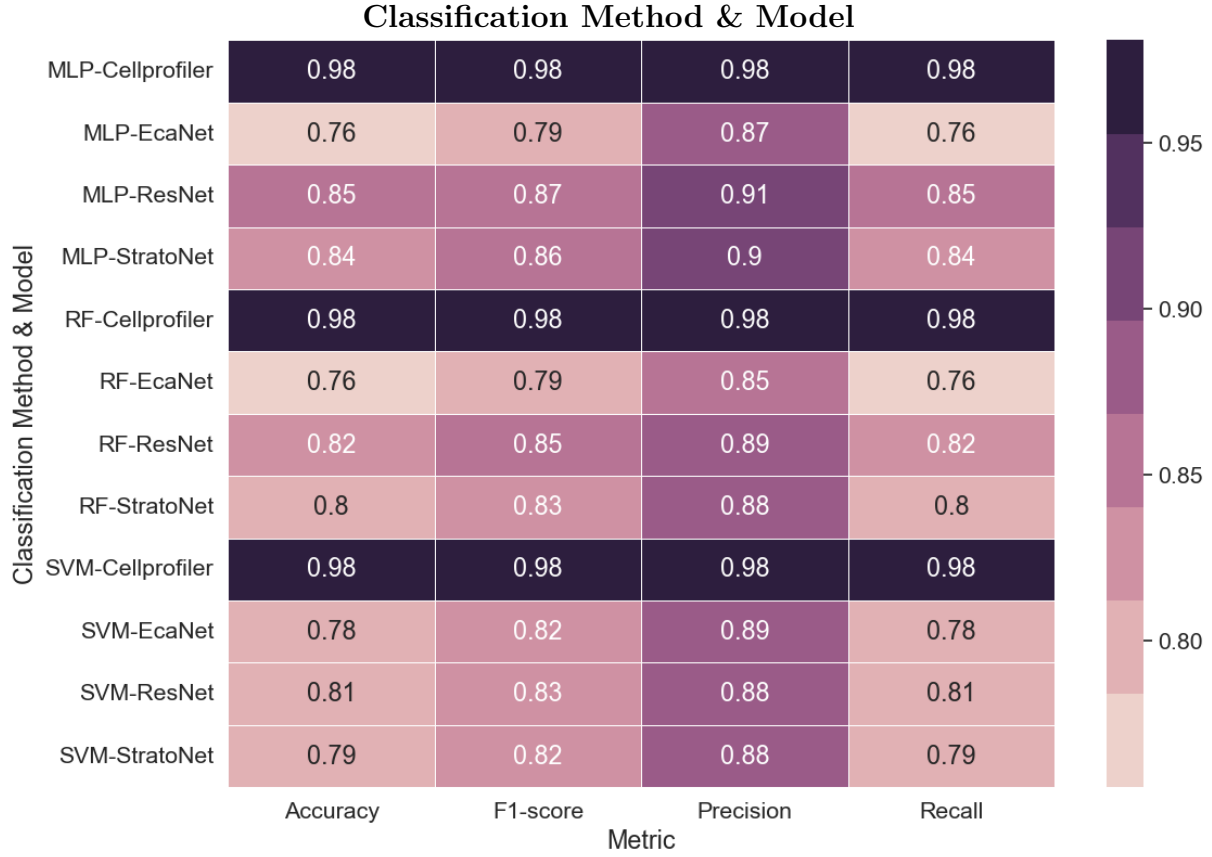
**Table 7:** Top-1 and Top-5 accuracy along with parameters count in millions for different neural network models.

### 6.2 High Content Screening Feature Extraction

The results for the feature extraction positions are visualized in Appendix D. The feature extraction position with the highest distance between controls for this dataset would be the Appendix D 14a convolution 3 from residual block 4. This intermediate feature extraction position was used for both **ResNet** and **ECANet**. For StratoNet, the later intermediate feature extraction positions convolution 4 from residual block 4 and convolution 4 from residual block 6 seen in Appendix D 14b, D 14c were used. They will be referred to as low (convolution 3 from residual block 4), medium (convolution 4 from residual block 4), and high (convolution 4 from residual block 6) level feature extraction.

### 6.2.1 Classification Methods

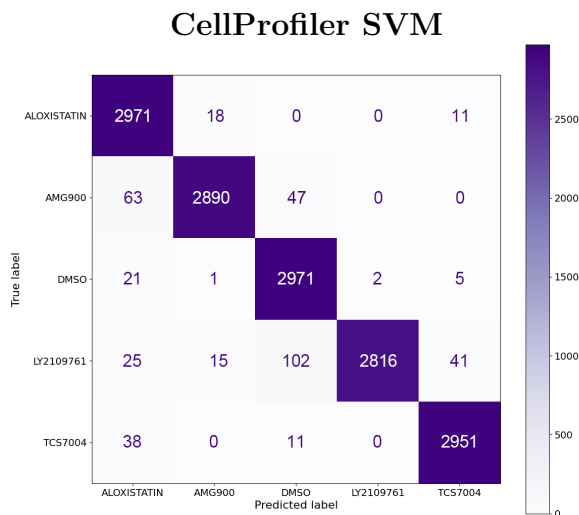
The second part of the results compares the StratoNet model to the HCS standard and the neural networks on which it is based, as previously mentioned. Figure 7 shows the results of CellProfiler, Resnet50, ECANet, and StratoNet. As previously discussed, all features from the aforementioned methods are classified using three distinct classification techniques. CellProfiler exhibits the highest performance on all four metrics for all classification methods. However, The neural network methods exhibit suboptimal performance yet remain above 75% on all metrics across all models.



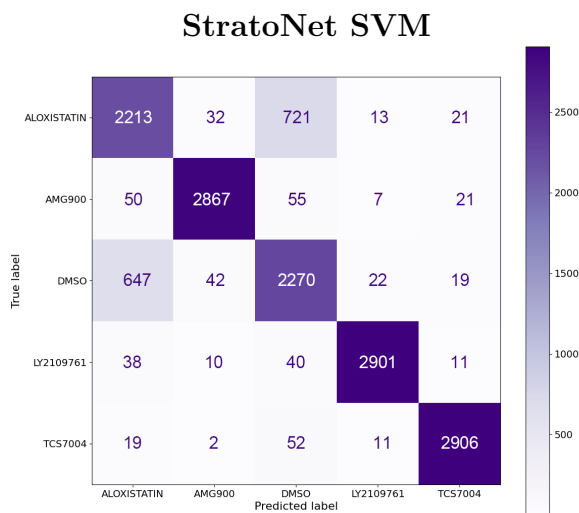
**Figure 7:** The classification results for CellProfiler [12], Resnet50 [28], ECANet [20], and StratoNet grouped by Multilayer Perceptrons (MLP), Random Forest (RF), and Support Vector Machines (SVM). To address the issue of class imbalance, macro averages have been employed.

## 6.2.2 Filtered Classification Methods

The class ALOXISTATIN is more challenging to classify in the **Jump-CP** dataset as it does not create an obvious phenotype. As shown in the confusion matrix in Figure 8, StratoNet misclassified DMSO and Aloxistatin. The neural networks performed worse, specifically in these classes, as seen in Appendix E. The confusion matrix in Figure 8 shows the performance of both CellProfiler and StratoNet for the SVM classification on the different classes. This test set is balanced for this visualization.



(a). The SVM confusion matrix for CellProfiler

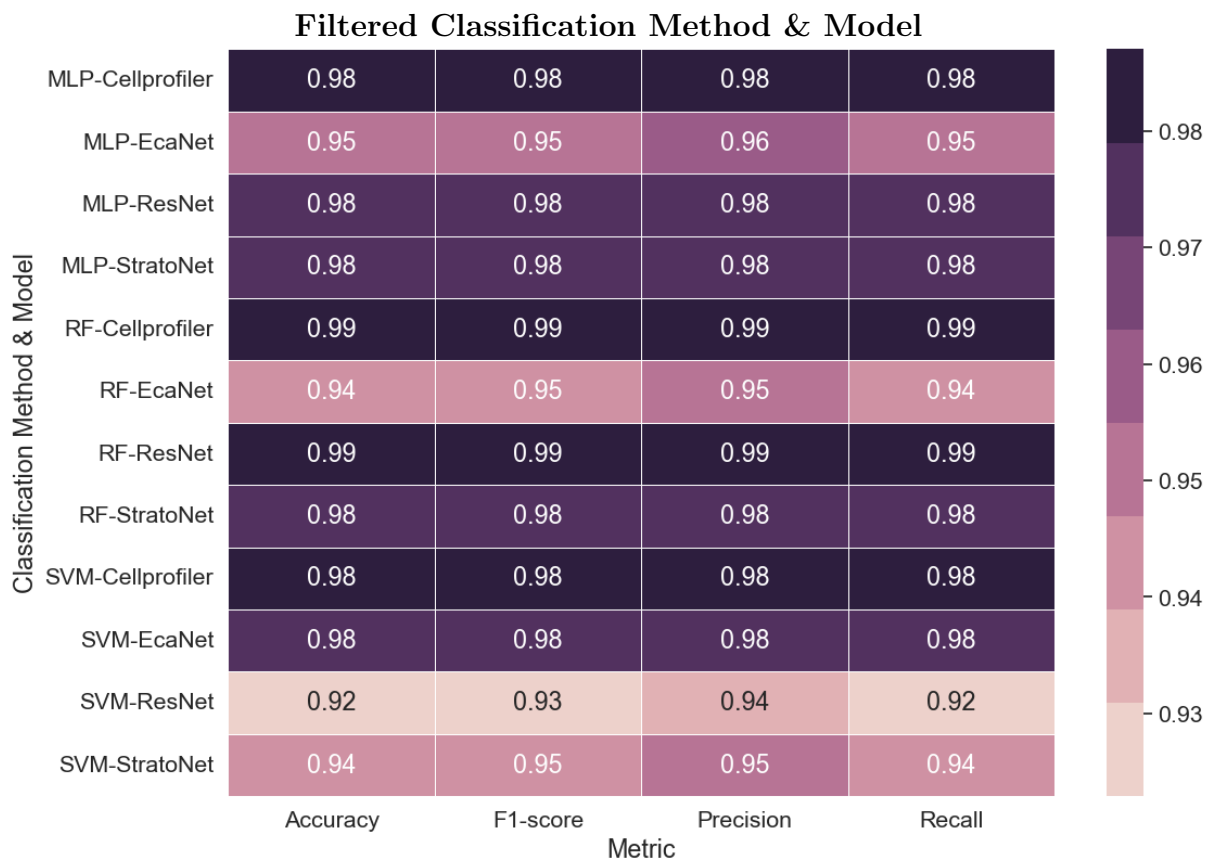


(b). The SVM confusion matrix for StratoNet

**Figure 8:** Showing the MLP confusion matrices for both CellProfiler and StratoNet for the five classes in the Jump-CP dataset



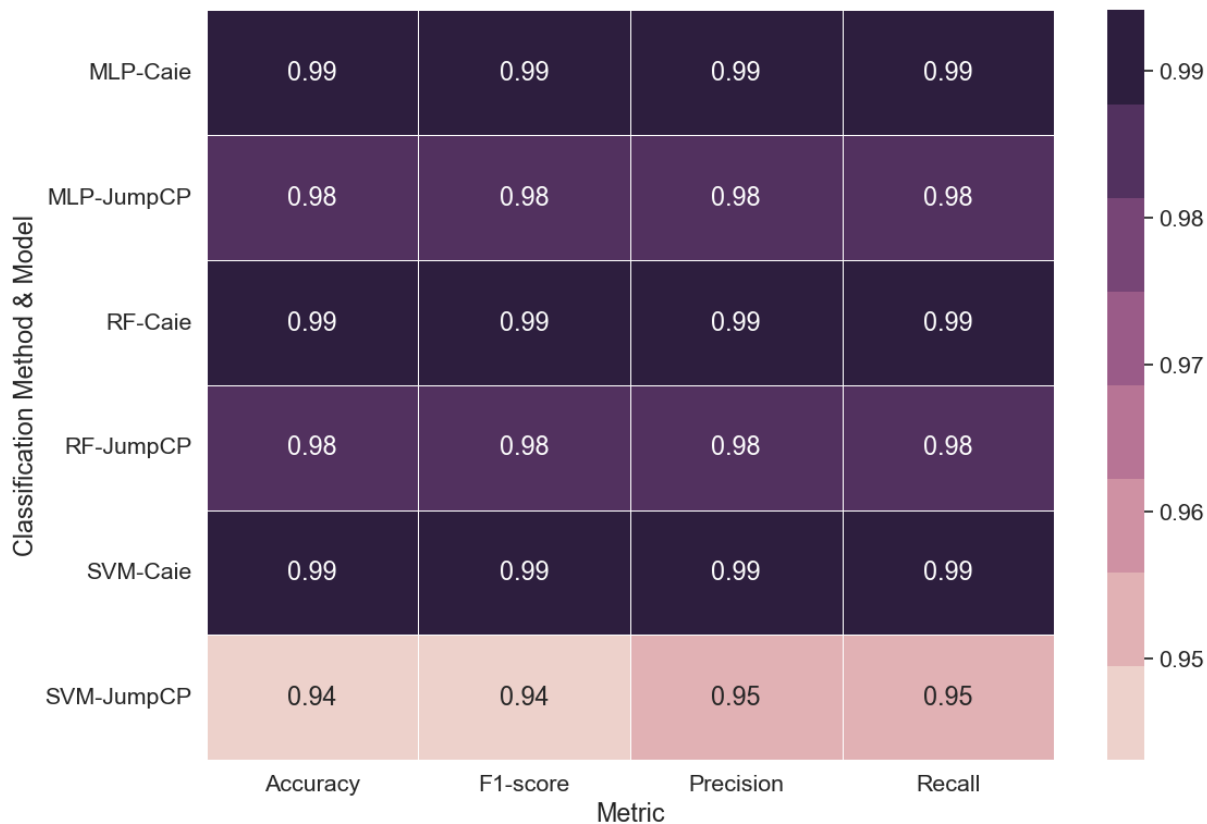
Overall, the neural networks underperformed, prompting the generation of an additional filtered result that excluded the Aloxistatin class from the dataset. The train and test set were filtered, and all classification methods were re-evaluated using the filtered dataset. The results of this filtered dataset are shown in Figure 9. It can be observed that all models exhibit optimal performance, exceeding 90%. Additionally, it can be observed that there is a difference in performance for the different classification methods, as evidenced by the comparison between ResNet SVM and ResNet MLP. CellProfiler demonstrated a notable improvement, increasing from 98% in Figure 7 to 99% in Figure 9 for the random forest classification, while the other methods for CellProfiler exhibited no improvement.



**Figure 9:** The filtered classification results for Cellprofiler [12], Resnet50 [28], Eca-Net [20], and StratoNet were grouped by Multilayer Perceptrons (MLP), Random Forest (RF), and Support Vector Machines (SVM). For this classification the reagent class ‘ALOXISTATIN’ was filtered out. To address the issue of class imbalance, weighted averages were employed.

### 6.2.3 Transferability of StratoNet

To showcase the versatility of the StratoNet model, we compare its performance on two different datasets: the JumpCP source 8 data (depicted in Figure 7) and the Caie dataset. The results are outlined in Figure 10, which also includes the performance of StratoNet on the JumpCP source 8 data.

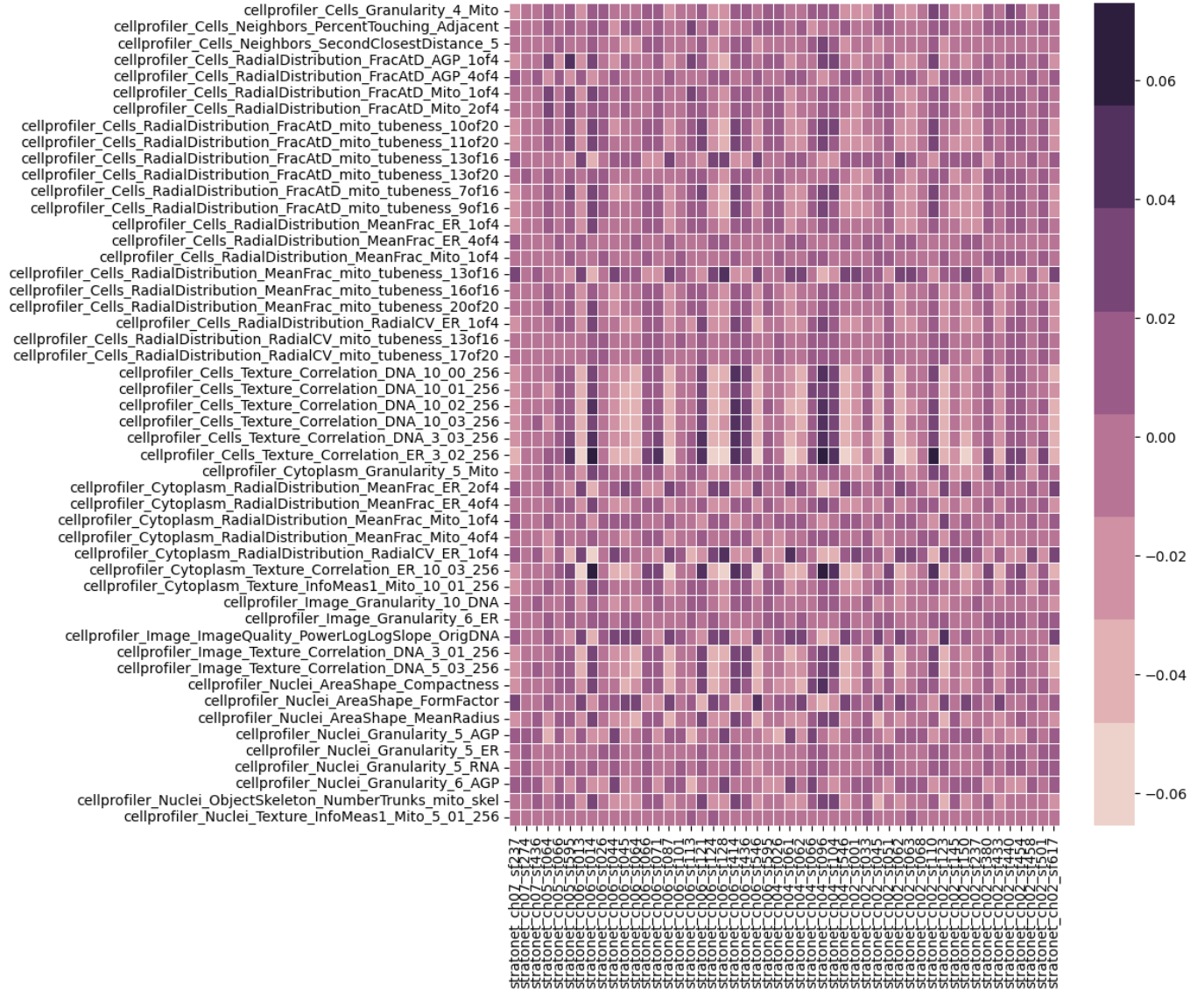


**Figure 10:** Accuracy scores for two different datasets with error folds tested at 10x cross validation

Figure 10 shows that StratoNet has similar performance on both datasets. This indicates that the model's performance is consistent across different data sources. Notably, these results are derived from the filtered **Jump-CP** dataset, which excludes the Aloxistatin class. The consistency could also be seen for all classification methods.

## 6.2.4 StratoNet Features vs CellProfiler Features

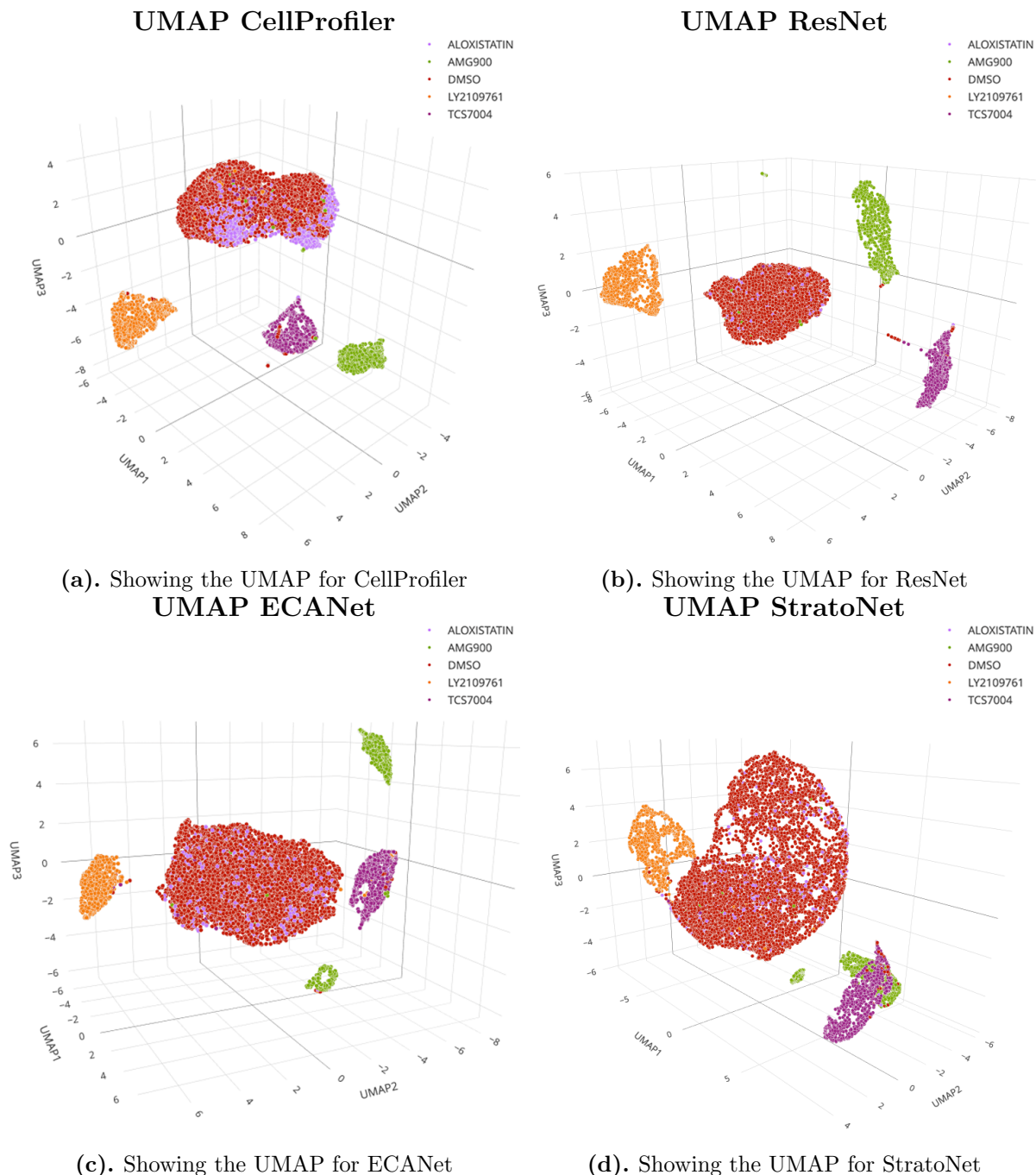
To compare the features for both CellProfiler and StratoFeatures, a correlation plot shows the correlation for the top 50 classification features for both methods. A correlation was calculated for a sample of all features, but no significant correlations were found. Therefore, the top 50 were visualized in Figure 11. Notice that the bar legend shows values in the range  $[-0.07, 0.07]$ .



**Figure 11:** Correlation plot showing the correlation between stratoFeatures and CellProfiler features

### 6.2.5 Feature Mappings

To conclude the results and provide a visual summary, **UMAP** plots are generated from the principal components of the data. These UMAPs offer an intuitive way to understand the distribution and clustering of the different classes within the datasets. The UMAPs highlight the distinct separation and relationships between the various chemical compounds by reducing the dimensionality as seen in Figure 12. These visualizations confirm the effectiveness of the feature extraction and model performance, illustrating clear groupings and separations that align with the expected biological and chemical distinctions. The **UMAP** plots are a powerful tool to conclude our analysis, providing clear evidence of the robustness and accuracy of the StratoNet model in handling complex **HCS** data.



**Figure 12:** UMAPs for the different methods on the Jump-CP dataset

## 7 Discussion

The outcomes of the StratoNet are not as impressive on the ImageNet dataset as those of both ResNet and ECANet, as evidenced by Table 7. In order to optimize the feature extraction process, the positions of the extractions were strategically positioned to maximize the performance of the JumpCP dataset, as illustrated in Appendix Figure D. This resulted in the last intermediate feature extraction occurring relatively early in the network, reducing the number of parameters to be trained from 25.6 million (ResNet) or 29.5 million (ECANet) to 8.5 million (StratoNet). In addition to the reduced number of trainable parameters resulting from the early feature extraction of the network, other potential causes must also be considered. Various network architectures with varying outcomes were tested during the StratoNet training process. Ultimately, the decision was made to adhere as closely as possible to the original network architecture, after which further variability was ruled out. The limited scope of experimentation was constrained by resource limitations, which may have affected the performance of StratoNet compared to the former. In total, approximately 50+ architectural modifications were implemented. The most stable and optimal performance was achieved when the original architecture was adhered to. However, the optimal hyperparameters have probably not yet been found for the StratoNet, but a lot has been scratched off. However, the last few percent would need much tuning to increase the accuracy further.

Deep learning feature extraction performance on HCS data appears promising, achieving results comparable to those obtained with CellProfiler as seen in Table 9. However, upon initial examination, the results appeared inferior, as illustrated in Table 7. However, upon closer examination, the results in Table 9 demonstrated that the approach yielded results nearly equivalent to those obtained with CellProfiler. These results were expected, given that the control ALOXISTATIN did not show any apparent phenotypic effect on the cells as displayed in Figure 8. Nevertheless, CellProfiler could still classify ALOXISTATIN correctly, which is remarkable.

The features extracted from the StratoNet were reproducible by design, but if someone tried to repeat this work and train their own network from scratch, the weights wouldn't be identical because of the random component and thus there would be variability. However, this variability is caused by the random component of the network and training and not by the user, the risk of bias is reduced. Another influence on the performance of neural networks could be the effect of downsampling and cropping the images for StratoFeatures (224x224) vs CellProfiler input (1024x1024). Downsampling carries the risk of information loss; other methods would be multi-based cropping, but to do this in a reproducible way would be challenging and could unnecessarily explode the feature space. Another way to avoid downsampling is to increase the input size. This could be done in this case, but then all the additional layers for all the neural networks would also have to be trained, and the ImageNet-1k dataset does not have the same image resolutions as Jump-CP. This additional training could lead to worse results. Furthermore, working with larger images also increases the computational complexity, which increases the runtimes from Appendix C to even more significant numbers.

During this thesis, it was found that reproducing specific results was more difficult than expected, even with the correct information. For example, reproducing the original ResNet results with the weights provided by Tensorflow was initially unsuccessful. The original paper stated that the input sizes used were 224x224. However, upon further inspection of the source of the Tensorflow ResNet-50 weights, it was found that the input sizes had been changed to 299x299. After fixing this problem, the same results were found. Therefore, when reusing and saving models, they should come with their respective usage information, otherwise they become useless.

## 8 Conclusion

This thesis confirms the results of previous work that neural networks contribute positively to the field of **HCS**. The StratoNet proposal performed similarly to other neural network solutions. However, it lacked classification of less susceptible phenotypic changes as did other neural networks. Also, the performance on the ImageNet dataset was suboptimal. However, it must be noted that the network size and, thus, the number of trainable parameters was reduced because of the feature extraction positions. Still, the performance was near optimal with one-third of the trainable parameters.

In addition to using StratoNet, channel-specific features were also shown to contribute to the correct classification of **MoA** classes. The disadvantage is that the feature space grows for each available channel. However, this does not outweigh the fact that each experiment with similar controls can now be compared if the unavailable features are filtered out.

The case is that for **Jump-CP** and the *Caie et al.* dataset, the same cellular component was targeted, the nucleus; however, two different stains were used: Hoechst vs. DAPI. Still, the performance was comparable between the two different datasets.

Although segmentation is an effective method for identifying objects in images, this thesis demonstrates that alternative solutions exist for **MoA** predictions. Segmentation is an essential but challenging method that, when unsuccessful, may result in missing data to incorrectly segmented cellular components and their associated features. StratoNet cannot create unintentional missing data because it quantifies all available images.

The first sub-research question was answered by showing that for each neural network, a feature set was quantified that was able to correctly classify the **MoA** classes as shown in Figures 7, and 9. The second sub-research question was answered by Figure 10, where the implemented methods of StratoNet showed that it could correctly classify different classes from different experiments while the model was static. The third and last sub-research question was answered by the comparisons in Figures 7 and 9, while also showing little correlation in Figure 11. The main research question is the conclusion of the above. StratoNet can quantify valuable features in the **HCS** field. The features are reproducible and cannot be changed by the end user.

This thesis introduced the StratoNet as a novel feature extraction network that produces effective reproducible features. Combined with state of the art methods, StratoNet showed great potential, although not fully optimal. StratoNet seems to be a possible addition to a regular **HCS** future experiment. It can be used on many different occasions, for example, in quality control and exploratory research, and as a complement to current quantification methods to capture even more variance in an image dataset. This work was a poster presentation at the Society for Laboratory Automation and Screening conference in 2024. [32]

### 8.1 Future work

A lot of work has been done within this thesis. However, many exciting things still exist to explore and implement. First, **ResNet** has now been chosen as the foundation for StratoNet. However, it can be replaced by any other **CNN**. Several other **CNNs** have been used for **HCS**, such as efficientNet and U-Net, to name a few. Further hyperparameter tuning can be performed. The intermediate feature extractions can be changed, or the end user can choose their intermediate layers. Further analysis could be done for domain invariance, as this thesis shows the performance from training the ImageNet-1K dataset to the Jump-CP dataset using StratoNet as the feature extraction model, but more domain datasets could be tested. Also, more challenging datasets could be tested using the StratoNet; for example, neuron datasets could be an exciting start to quantify. Neuron datasets are typically more complex to segment as neurons overlap in a 3 dimensional space. Altogether, this work has demonstrated its potential as a foundational component for future exploratory **HCS** experiments, paving the way for more advanced and comprehensive studies in this field.



# References

- [1] M. Schlander, K. Hernandez-Villafuerte, C.-Y. Cheng, J. Mestre-Ferrandiz, and M. Baumann, "How much does it cost to research and develop a new drug? a systematic review and assessment," *PharmacoEconomics*, vol. 39, no. 11, pp. 1243–1269, Nov 2021. [Online]. Available: <https://doi.org/10.1007/s40273-021-01065-y>
- [2] O. Z. Kraus and B. J. Frey, "Computer vision for high content screening," *Critical Reviews in Biochemistry and Molecular Biology*, vol. 51, no. 2, pp. 102–109, 2016.
- [3] A. Lin, C. J. Giuliano, A. Palladino, K. M. John, C. Abramowicz, M. L. Yuan, E. L. Sausville, D. A. Lukow, L. Liu, A. R. Chait, Z. C. Galluzzo, C. Tucker, and J. M. Sheltzer, "Off-target toxicity is a common mechanism of action of cancer drugs undergoing clinical trials," *Sci Transl Med*, vol. 11, no. 509, Sep. 2019.
- [4] J. Sun, A. Tárnok, and X. Su, "Deep learning-based single-cell optical image studies," *Cytometry Part A*, vol. 97, no. 3, pp. 226–240, 2020. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cyto.a.23973>
- [5] G. Tian, P. J. Harrison, A. P. Sreenivasan, J. Carreras-Puigvert, and O. Spjuth, "Combining molecular and cell painting image data for mechanism of action prediction," *Artificial Intelligence in the Life Sciences*, vol. 3, p. 100060, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2667318523000041>
- [6] J. O. Cross-Zamirski, E. Mouchet, G. Williams, C.-B. Schönlieb, R. Turkki, and Y. Wang, "Label-free prediction of cell painting from brightfield images," *Sci Rep*, vol. 12, no. 1, p. 10001, Jun. 2022.
- [7] A. Kensert, P. J. Harrison, and O. Spjuth, "Transfer learning with deep convolutional neural networks for classifying cellular morphological changes," *SLAS Discov*, vol. 24, no. 4, pp. 466–475, Jan. 2019.
- [8] C. McQuin, A. Goodman, V. Chernyshev, L. Kametsky, B. A. Cimini, K. W. Karhohs, M. Doan, L. Ding, S. M. Rafelski, D. Thirstrup, W. Wiegraebe, S. Singh, T. Becker, J. C. Caicedo, and A. E. Carpenter, "CellProfiler 3.0: Next-generation image processing for biology," *PLoS Biol*, vol. 16, no. 7, p. e2005970, Jul. 2018.
- [9] M. Devices, "In carta analysis software," *Application Note*, vol. 1, no. 1, pp. 1–20, 2020. [Online]. Available: <https://www.moleculardevices.com/sites/default/files/en/assets/user-guide/dd/img/in-carta-analysis-software-sinap-instructions-user-manual.pdf>
- [10] Thermofisher, "Analysis of cancer spheroids through high-throughput screening assays," *Application Note*, vol. 1, no. 1, pp. 1–6, 2020. [Online]. Available: <https://assets.thermofisher.com/TFS-Assets/BID/Application-Notes/analysis-cancer-spheroids-high-throughput-screening-assays-app-note.pdf>
- [11] A. Niederlein, "Image analysis in high content screening," *Combinatorial Chemistry High Throughput Screening*, vol. 12, no. 9, pp. 899–907, 2009. [Online]. Available: <http://www.eurekaselect.com/article/15198>
- [12] D. R. Stirling, M. J. Swain-Bowden, A. M. Lucas, A. E. Carpenter, B. A. Cimini, and A. Goodman, "Cellprofiler 4: improvements in speed, utility and usability," *BMC Bioinformatics*, vol. 22, no. 1, p. 433, Sep 2021. [Online]. Available: <https://doi.org/10.1186/s12859-021-04344-9>

- [13] J. Z. Sexton, R. Fursmidt, M. J. O'Meara, W. Omta, A. Rao, D. A. Egan, and S. A. Haney, "Machine learning and assay development for image-based phenotypic profiling of drug treatments," *Assay Guidance Manual [Internet]*, 2023.
- [14] D. R. Wong, D. J. Logan, S. Hariharan, R. Stanton, D.-A. Clevert, and A. Kiruluta, "Deep representation learning determines drug mechanism of action from cell painting images," *Digital Discovery*, vol. 2, no. 5, pp. 1354–1367, 2023.
- [15] L. Alzubaidi, J. Bai, A. Al-Sabaawi, J. Santamaría, A. S. Albahri, B. S. N. Al-dabbagh, M. A. Fadhel, M. Manoufali, J. Zhang, A. H. Al-Timemy, Y. Duan, A. Abdullah, L. Farhan, Y. Lu, A. Gupta, F. Albu, A. Abbosh, and Y. Gu, "A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications," *Journal of Big Data*, vol. 10, no. 1, p. 46, Apr 2023. [Online]. Available: <https://doi.org/10.1186/s40537-023-00727-2>
- [16] S. N. Chandrasekaran, J. Ackerman, E. Alix, D. M. Ando, J. Arevalo, M. Bennion, N. Boisseau, A. Borowa, J. D. Boyd, L. Brino *et al.*, "Jump cell painting dataset: morphological impact of 136,000 chemical and genetic perturbations," *Biorxiv*, pp. 2023–03, 2023.
- [17] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
- [18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.
- [19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [20] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: Efficient channel attention for deep convolutional neural networks," 2020.
- [21] M. Hofmarcher, E. Rumetshofer, D.-A. Clevert, S. Hochreiter, and G. Klambauer, "Accurate prediction of biological assays with high-throughput microscopy images and convolutional networks," *Journal of Chemical Information and Modeling*, vol. 59, no. 3, pp. 1163–1171, 2019. [Online]. Available: <https://doi.org/10.1021/acs.jcim.8b00670>
- [22] B. Zhou, A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," *CoRR*, vol. abs/1512.04150, 2015. [Online]. Available: <http://arxiv.org/abs/1512.04150>
- [23] F. Laakom, K. Chumachenko, J. Raitoharju, A. Iosifidis, and M. Gabbouj, "Learning to ignore: rethinking attention in cnns," *CoRR*, vol. abs/2111.05684, 2021. [Online]. Available: <https://arxiv.org/abs/2111.05684>
- [24] W. A. Omta, R. G. van Heesbeen, R. J. Pagliero, L. M. van der Velden, D. Lelieveld, M. Nellen, M. Kramer, M. Yeong, A. M. Saeidi, R. H. Medema *et al.*, "Hc stratominer: A web-based tool for the rapid analysis of high-content datasets," *Assay and drug development technologies*, vol. 14, no. 8, pp. 439–452, 2016.
- [25] A. E. Carpenter, T. R. Jones, M. R. Lamprecht, C. Clarke, I. H. Kang, O. Friman, D. A. Guertin, J. H. Chang, R. A. Lindquist, J. Moffat, P. Golland, and D. M. Sabatini, "CellProfiler: image analysis software for identifying and quantifying cell phenotypes," *Genome Biol*, vol. 7, no. 10, p. R100, Oct. 2006.



- [26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
- [27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.
- [28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.
- [29] P. D. Caie, R. E. Walls, A. Ingleston-Orme, S. Daya, T. Houslay, R. Eagle, M. E. Roberts, and N. O. Carragher, "High-content phenotypic profiling of drug response signatures across distinct cancer cells," *Molecular cancer therapeutics*, vol. 9, no. 6, pp. 1913–1926, 2010.
- [30] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.
- [31] "Compute – Amazon EC2 Instance Types – AWS — aws.amazon.com," <https://aws.amazon.com/ec2/instance-types/>, [Accessed 25-04-2024].
- [32] e. a. Michel van Elswijk, Maria Roa Oyaga, "Cell painting in pictures and numbers: An integrated image and data analytics platform for accelerating drug discovery." poster presented at SLAS 2024 conference, Boston, America.

# Appendix

## A Required packages

The figure lists the requisite packages for the completion of all tasks associated with this work. It should be noted that the inclusion of version numbers is not mandatory, except for those explicitly indicated.

| Package                |
|------------------------|
| tensorflow             |
| keras                  |
| numpy                  |
| pandas==2.0.3          |
| Pillow                 |
| pyarrow                |
| scikit-image           |
| scikit-learn           |
| scipy==1.10.1          |
| tiffle                 |
| opencv-python          |
| fsspec                 |
| s3fs                   |
| mysql-connector-python |
| psutil                 |
| seaborn                |

**Table 8:** Required Packages

## B Training StratoNet

Below an algorithm box is displayed which depicts the training procedure for StratoNet. This training procedure was inspired by both ResNet, and EcaNet methods [28, 20].

---

**Algorithm 2** Training StratoNet

---

```
1: Initialize model with StratoNet architecture, set number of residual blocks
2: Initialize optimizer set momentum to 0.99 (e.g., SGD)
3: Initialize loss function (e.g., Categorical Cross Entropy )
4: Initialize learning rate scheduler and learning rate reducer
5: Load dataset in separate training and validation sets
6: Define number of epochs (e.g., 100) and batch size (e.g., 128)
7: Define image size (e.g., (224, 224, 3))
8: Print model summary, training size
9: for epoch in range(epochs) do
10:   Set model to training mode
11:   for each batch in training set do
12:     Forward pass:
13:     Predict the outputs from the model using the batch data.
14:     Compute the difference between the predicted outputs and the actual labels
    (e.g. Loss)
15:     Backward pass:
16:     Compute gradients by means of backpropagation
17:     Use optimizer to Update model parameters
18:     Update training loss and accuracy
19:   end for
20:   Set model to evaluation mode
21:   for each batch in validation set do
22:     Load batch data and labels
23:     Forward pass:
24:     Predict outputs from model using batch data
25:     Compute loss between predicted outputs and actual labels
26:     Update validation loss and accuracy
27:   end for
28:   Update learning rate using scheduler
29:   Print epoch, training loss, training top-1 accuracy, training top-5 accuracy valida-
    tion loss, validation top-1 accuracy, validation top-5 accuracy
30:   if validation top-1 accuracy improves then
31:     Save model checkpoint
32:   end if
33: end for
34: Return trained model
```

---

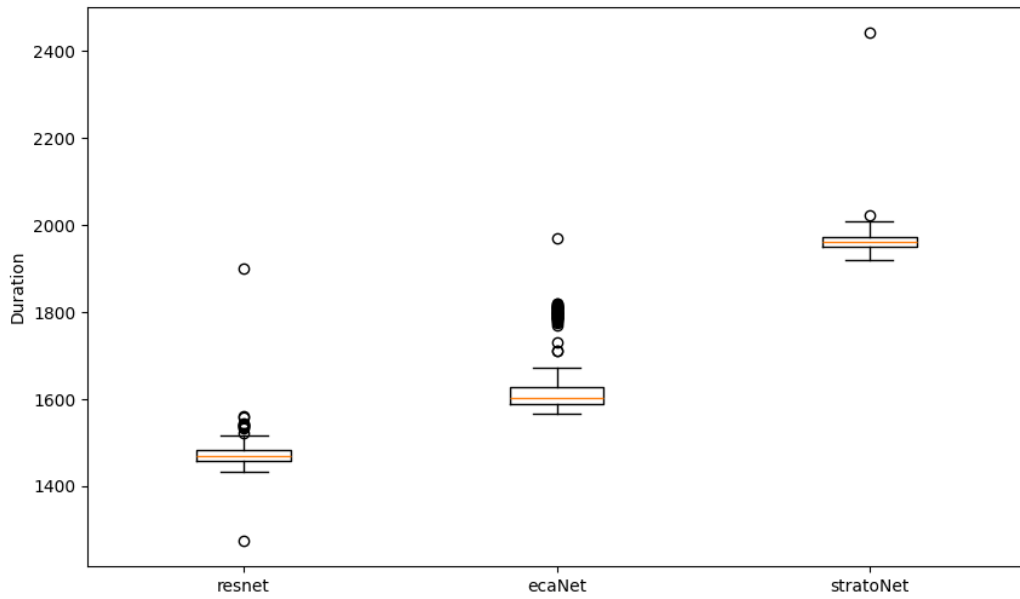
## C Performance in time

while not the principal objective of this study, the time required for feature extraction is acquired and reported in the following Table and figure.

| Approach          | CPU %   |      | Memory % |      | Realtime<br>(hours) |
|-------------------|---------|------|----------|------|---------------------|
|                   | Average | Max  | Average  | Max  |                     |
| Cellprofiler [12] | N/A     | N/A  | N/A      | N/A  | N/A                 |
| CPUltra [12]      | 24.2    | 100  | 22.9     | 51   | 51.4                |
| Resnet50 [28]     | 13.0    | 62.4 | 15.2     | 34.0 | 85.5                |
| Eca-Net [20]      | 16.8    | 95.2 | 15.2     | 31.9 | 98.5                |
| StratoNet         | 19.9    | 98.1 | 14.4     | 34.1 | 117.8               |

**Table 9:** The performance results of the feature extraction process for various methods

### Real-time performance of feature extraction methods

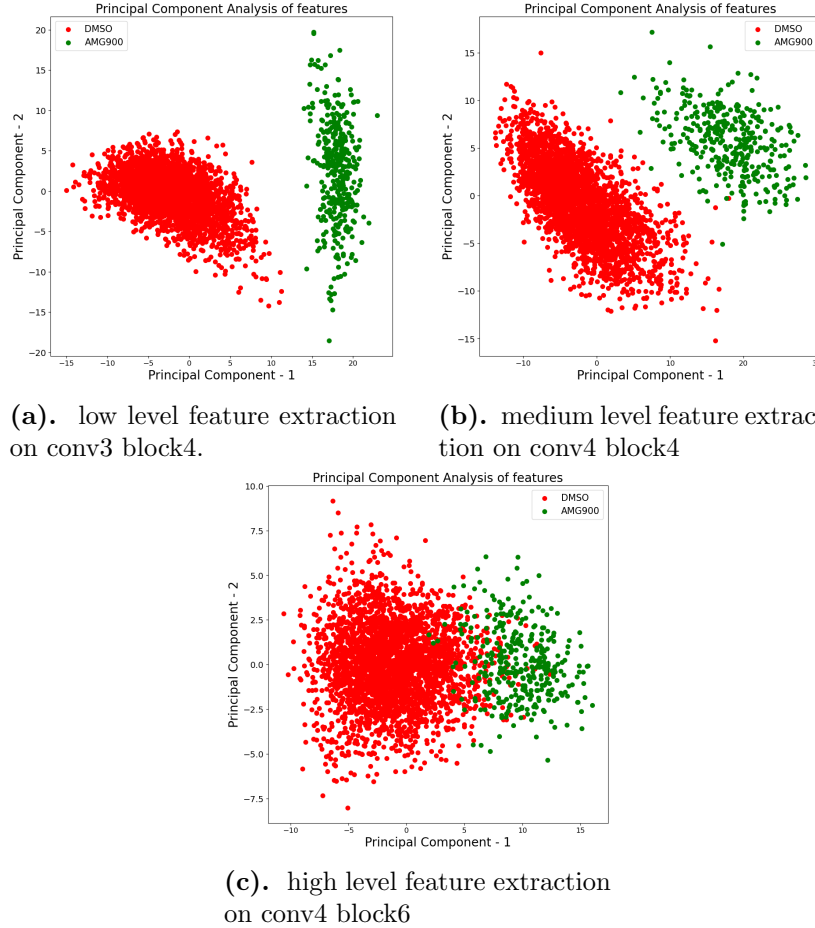


**Figure 13:** Boxplot showing the real-time performance of different feature extraction methods, showing the distribution, median, and variability of processing times

## D Intermediate Feature Extraction positions

The intermediate feature extraction positions in ResNet50 that demonstrated the most effective separation between positive and negative classes in the Jump-CP dataset are highlighted in the figure below.

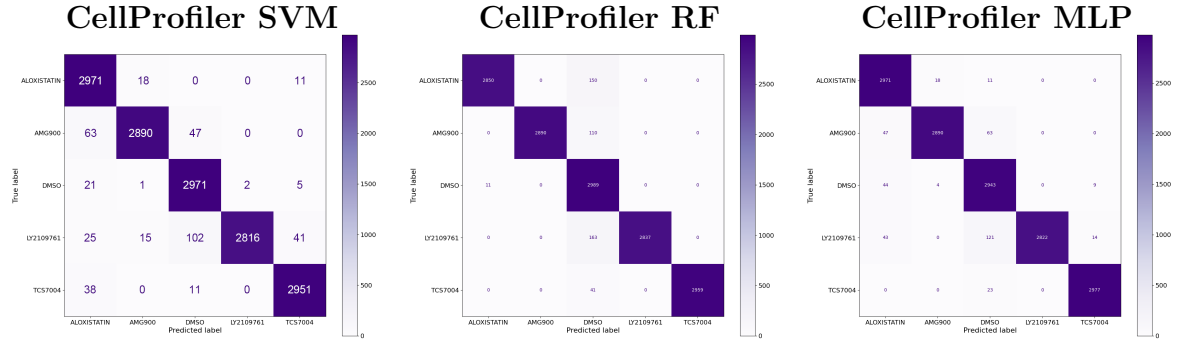
### Intermediate feature extractions at different positions in ResNet50



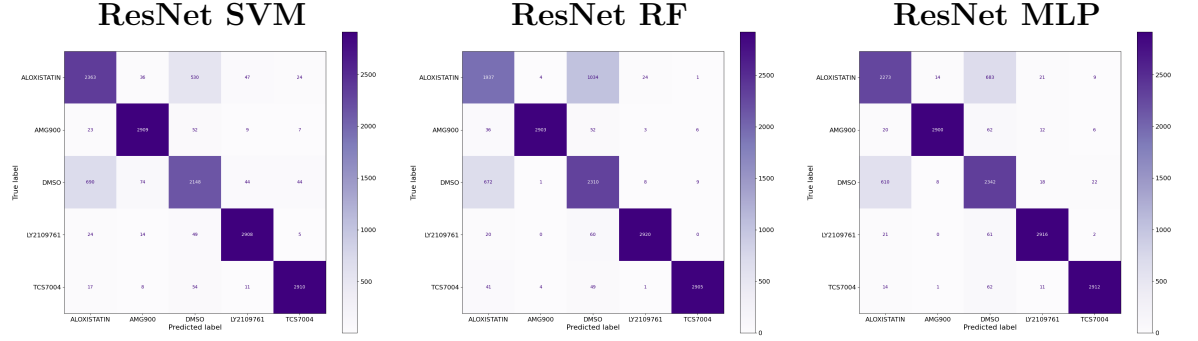
**Figure 14:** ResNet-50 showing the accuracy for 3 different intermediate feature extractions considering the positive and negative control. **a)** Illustrates the low-level feature extraction, which enables the separation of the positive and negative controls. **b)** displays the medium-level feature extraction of the positive and negative controls, while **c)** shows the high-level feature extraction of the positive and negative controls.

## E Individual confusion matrixes for all models

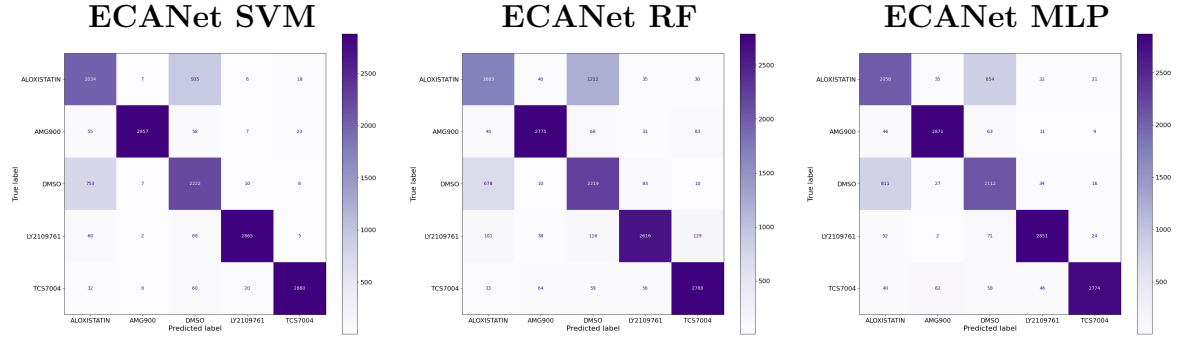
The subsequent page presents the confusion matrices for all trained models, which have been organized in a clear and concise manner.



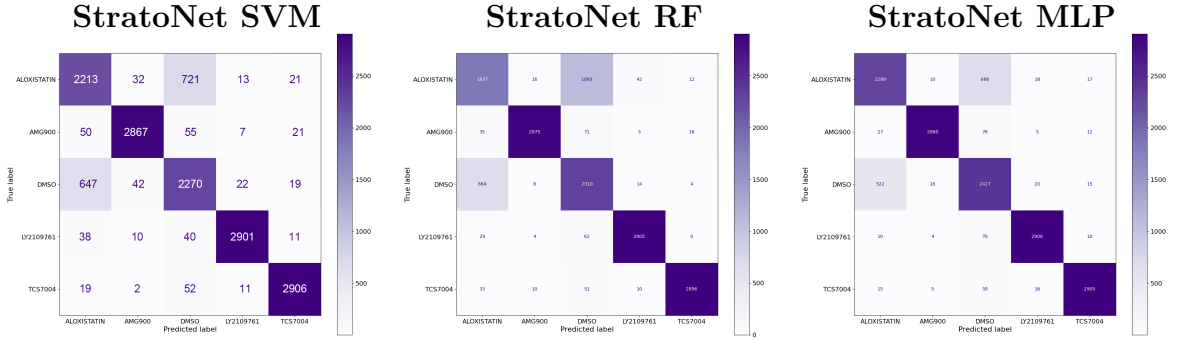
(a). The SVM confusion matrix for CellProfiler (b). The RF confusion matrix for CellProfiler (c). The MLP confusion matrix for CellProfiler



(d). The SVM confusion matrix for ResNet (e). The RF confusion matrix for ResNet (f). The MLP confusion matrix for ResNet



(g). The SVM confusion matrix for ECANet (h). The RF confusion matrix for ECANet (i). The MLP confusion matrix for ECANet



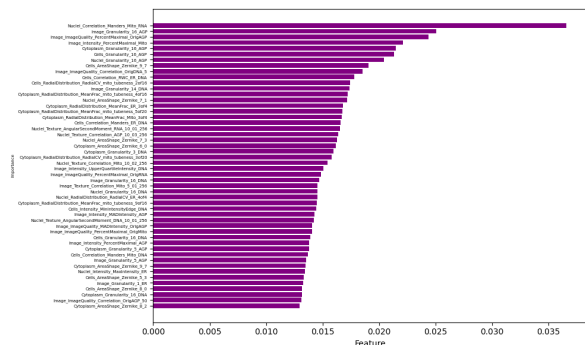
(j). The SVM confusion matrix for StratoNet (k). The RF confusion matrix for StratoNet (l). The MLP confusion matrix for StratoNet

**Figure 15:** Showing the MLP confusion matrices for CellProfiler, ResNet, ECANet, and StratoNet for the five classes in the Jump-CP dataset

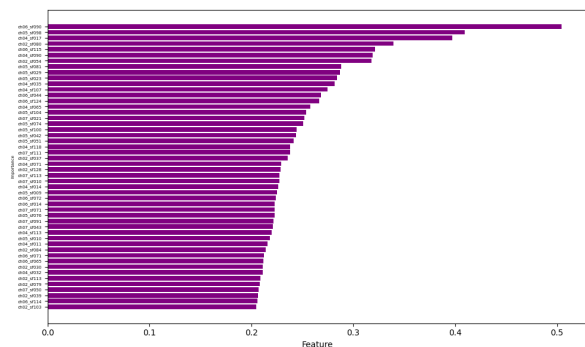
## **F    Feature importance for all models**

The subsequent page presents the Feature importance for all trained models, which have been organized in a clear and concise manner.

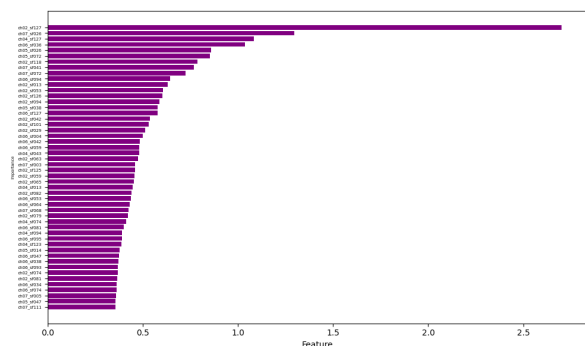
**CellProfiler Feature Importance SVM**



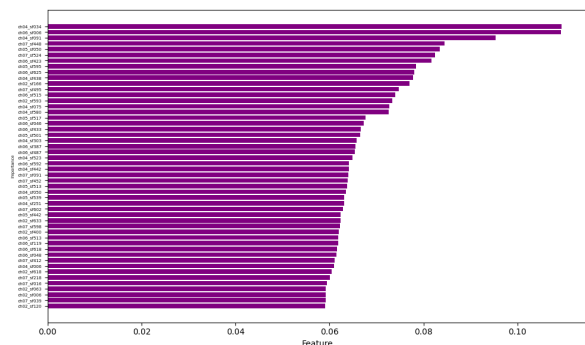
**(a). Showing the CellProfiler features importance RestNet Feature Importance SVM**



**(c). Showing the ResNet features importance ECANet Feature Importance SVM**

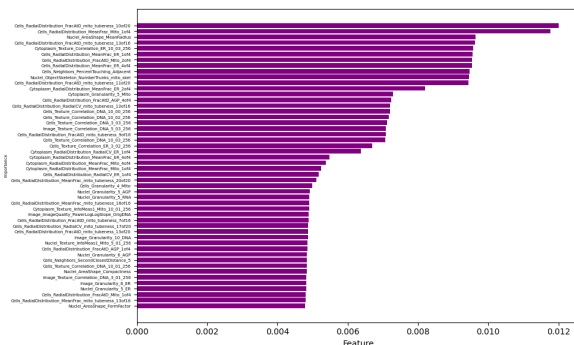


**(e). Showing the ECANet features importance StratoNet Feature Importance SVM**

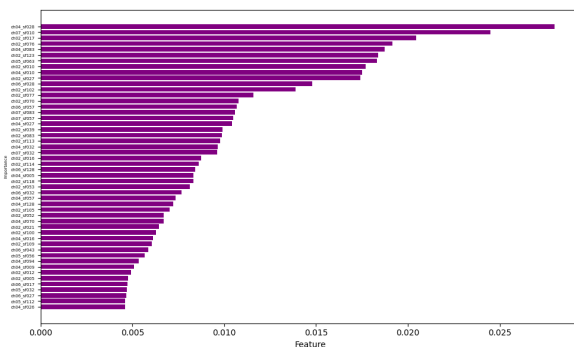


**(g). Showing the StratoNet features importance**

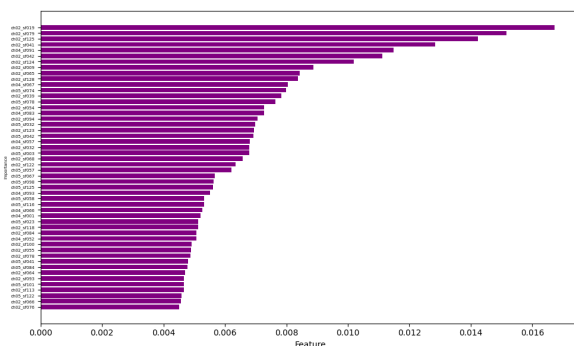
**CellProfiler Feature Importance RF**



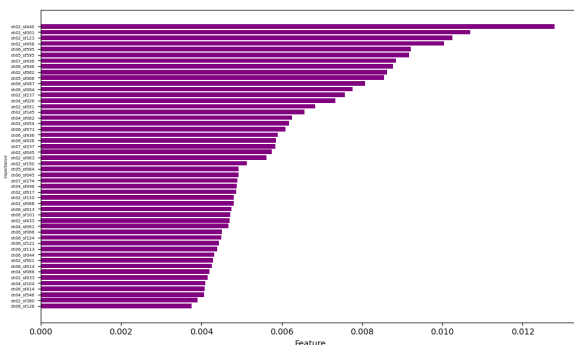
**(b). Showing the CellProfiler features importance RestNet Feature Importance RF**



**(d). Showing the ResNet features importance ECANet Feature Importance RF**



**(f). Showing the ECANet features importance StratoNet**



**(h). Showing the StratoNet features importance**

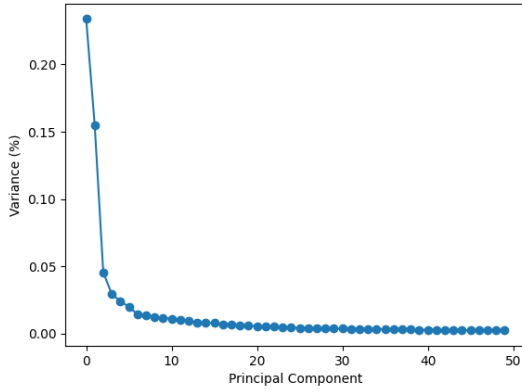
**Figure 16**



## G PCA Variances

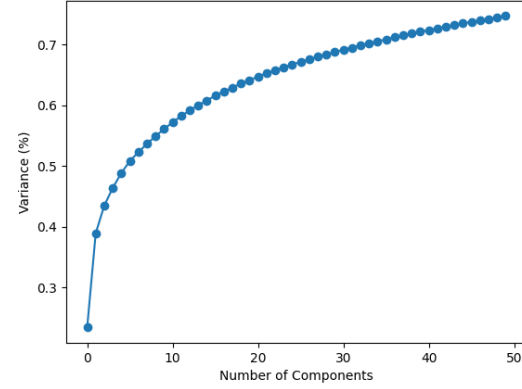
The subsequent page presents the variance captured for the models, organized in a clear and concise manner.

**ResNet PCA variance per component**



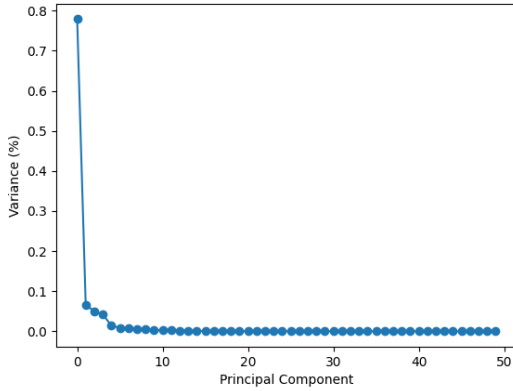
(a). Showing the ResNet variance per component

**ResNet PCA Cumulative Variance**



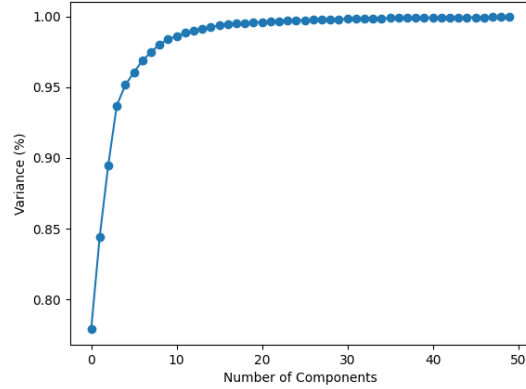
(b). Showing the ResNet PCA Cumulative Variance

**ECANet PCA variance per component**



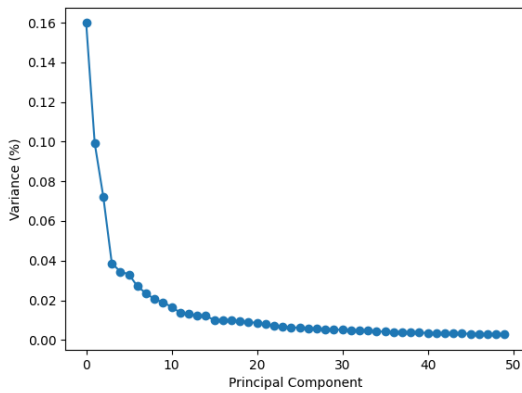
(c). Showing the ECANet variance per component

**ECANet PCA Cumulative Variance**



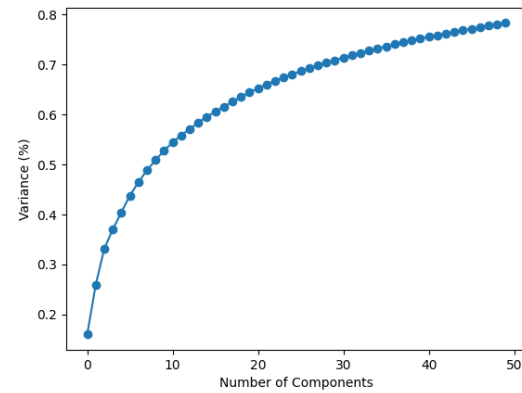
(d). Showing the ECANet Cumulative Variance

**StratoNet PCA variance per component**



(e). Showing PCA variance per component

**StratoNet PCA Cumulative Variance**



(f). Showing the StratoNet Cumulative Variance

**Figure 17**