Automatic High Content Screening Using Deep Learning

by

© *Farhad Mohammad Kazemi*

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the

requirements for the degree of

Master of *Science*

Department of *Computer Science*

Memorial University of Newfoundland

December 2018

St. John's Newfoundland

Abstract

Recently, deep learning algorithms have been used with success in a variety of domains. Deep learning has proven to be a very helpful tool for discovering complicated structures in high-dimensional and big datasets. In this work, five deep learning models inspired by AlexNet, VGG, and GoogleNet are developed to predict mechanism of actions (MOAs) based on phenotypic screens of a number of cells in dimly lit and noisy images. We demonstrate that our models can predict the MOA for a compendium of drugs that alter cells through single cell or cell population views without any segmentation and feature extraction steps. According to these results, our models do not need to fully realize single-cell measurements to profile samples because they use the morphology of specific phenomena in the cell population samples.

We used an imbalanced High Content Screening big dataset to predict MOAs with the main goal of understanding how to work properly with deep learning algorithms on imbalanced datasets when sampling methods, like Oversampling, Undersampling, and Synthetic Minority Over-sampling (SMOTE) algorithms are used for balancing the dataset. Based on our findings, it is now clear that the SMOTE sampling algorithm must be part of the deep learning algorithms when confronting imbalanced datasets.

High Content Screening technologies have to deal with screening thousands of cells to provide a number of parameters for each cell, such as nuclear size, nuclear morphology, DNA replication, etc. The success of High Content Screening (HCS) systems depends on automatic image analysis. Recently, deep learning algorithms have overcome object recognition challenges on tasks with a single centered object per image. Present deep learning algorithms have not been applied to images that

include multiple specific complex objects, such as microscopic images of many objects such as cells in these images.

Key Words: machine learning, deep learning, Artificial Neural Network, Data Science, Big Data, High Content Screening, High Content Analysis, Drug Discovery, Predict Phenomena, Predictive Analysis, Bioinformatics, AlexNet, VGG, GoogleNet, Inception, Oversampling, Undersampling, SMOTE, Anomaly Detection

Acknowledgements

This thesis was developed on Compute Canada HPC clusters. This work was enabled in part by support provided by ACENET, Calcul Quebec, WestGrid and Compute Canada (www.computecanada.ca). I would first like to thank my supervisors Prof. Wolfgang Banzhaf and Prof. Minglun Gong at Memorial University of Newfoundland and Labrador. They were always available whenever I faced a challenge or had a question about my research. They have guided me and encouraged me to carry on through these years. Thank you for guiding and supporting me.

My sincere thanks must also go to the members of my thesis advisory and exam committee: Dr. Andrew Vardy and Dr. Oscar Meruvia-Pastor. They generously gave their time to offer me valuable comments toward improving my work.

One person at the Biochemistry Department and Computational Chemical Biology & Drug Design Group, School of Pharmacy at Memorial University of Newfoundland deserves a very special word of gratitude: Zahed Khatooni who has given me valuable support in the beginning of this project.

I am most grateful to the collaborators for lending me their expertise and intuition to my scientific and technical problems: Dr. Oliver Stueker and other computational scientists at ACENET and Compute Canada.

I am thankful to my friends for all the support they provided, Dr Shahab Shamshirband at NTNU, Norwegian University of Science and Technology, and Zili Yi.

Also, I must express my very profound gratitude to my parents and to my lovely brother and dear Maryam for providing me with unfailing support throughout my years of study. This work would not have been possible without them. The last word

goes for Baran, my daughter, who has been the light of my life for these years and who has given me the extra strength and motivation to get things done. This work is dedicated to her.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

High Content Screening (HCS) is a technology applied in biological studies and drug discovery to recognize objects such as molecules or RNAi that change the phenotype of a cell [27, 65].

1.1 Problem Statement

HCS is used for phenotypic screening of cells, see Figure 1.1. Phenotypic variations sometimes can increase or decrease the size or quantity of cellular components such as proteins or modify the appearance and morphology of a cell.

Recent microscopy and high performance computing technologies have rapidly advanced the development of high-throughput image-based techniques. The success of HCS systems is dependent on automated image analysis. These methods deal with the screening of thousands of cells to provide a number of parameters for each cell, such as DNA replication, nuclear size, nuclear morphology, etc. Analyzing this high-dimensional problem requires machine learning algorithms. We believe that as

these parameters are high dimensional, deep learning algorithms can be useful to help classify, cluster and visualize cells in High Content Screening applications.

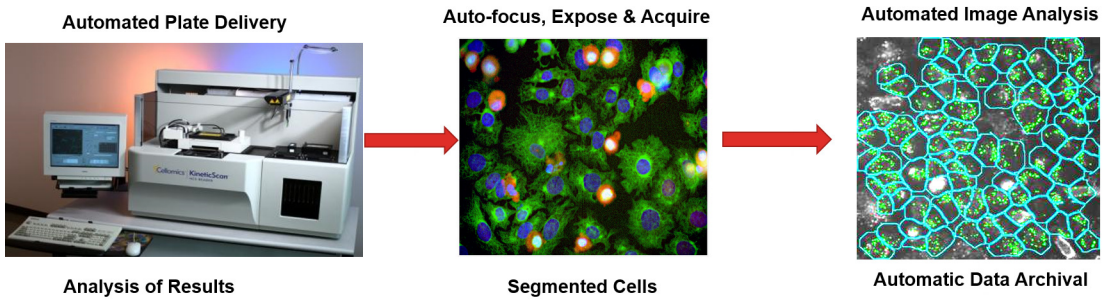

Figure 1.1: High Content Screening system overview

1.2 The Contribution of this Thesis

Analyzing large amounts of microscopy data is a challenging problem. Cell images have different features, where most of the cells are characterized by noisy and cluttered background patterns that make segmentation and automatic recognition of cells very problematic.

1- Recent microscopy and high performance computing technologies have made advancement of high throughput screening systems that are used in drug discovery, biology, and chemistry. Most subtle features such as DNA replication, nuclear size, and nuclear morphology can be observed in each image precisely by using High Content Analysis (HCA) methods. To solve this high-dimensional problem, biologists feed single cell information to machine learning algorithms such as K-nearest neighbors, support vector machines, or random forests. According to a recent study [61], about 70% of the papers on HCA experiments published in *Science, Nature, Cell, and the Proceedings of the National Academy of Sciences* from 2000 to 2012 only used one or two of the cells measured features and less than 15% used more than six. As a result, about 85% of the research work in HCA underutilized potentially valuable information that might have helped to speed up early-stage drug discovery [61]. In this project, we are interested in exploring cutting-edge machine learning algorithms developed in the field of deep learning to solve high-dimensional data problems with thousands of measured features.

2- Recently, deep learning algorithms have overcome object recognition challenges. These tasks normally have a single centered object per image and current algorithms have not directly been tested on images with populations of objects (for example, the

population of cells in microscopy images). Generally, there are a number of objects such as cells in microscopy images. In this project we provide deep learning models based on populations of objects (cells) and a single cell view in order to predict MOAs of drugs without using any segmentation or feature extraction steps.

3- HCS, when formulated as a problem in machine learning, is usually considered as an object recognition task. Cell images have different characteristics. They are dimly lit and noisy. The main purpose of this project is to predict a label for each cell or a population of cells, based on poor lighting and noisy images containing a single cell or a population of cells.

4- In High Content Screening object identification, the natural biological variation amongst cell populations mostly provides a variety of phenotypes. This subject makes some phenotypes difficult to recognize and complicates the labeling task, particularly phenotypes that have low penetrance.

5- Although image samples provide a number of benefits and details, analyzing huge amounts of microscopy data is a challenging problem [32, 46]. Manual scoring of complex patterns and phenotypes such as cell protein localizations is hard and specialist cell biologists in various groups across the world score images manually, but variability between experts decrease accuracy [9, 66]. Computational profiling or classifying phenotypes is a substitute to manual evaluation, as was done by Chong et al. [17](2015). Most high content screens have only used a handful of parameters from each sample, not all parameters [17, 61, 62]. HCS needs experts to apply cutting-edge machine learning algorithms to classify or cluster cellular phenotypes that can not be assessed manually [1]. It is clear that a High Content Screening system involves large quantities of image data which makes it really expensive for us to detect and recog-

nize manually. The generalization ability of traditional methods is restricted when they encounter images with different spatial information because they rely mostly on previous knowledge. These algorithms are ineffective for Big Data since they involve a massive number of operations for estimating the priors. Moreover, traditional classifiers are impractical for real-time applications since they are slow when used on Big Data.

To summarize, the contribution of this study will be the provision of a High Content Screening system by deep learning algorithms to analyze HCS images in order to solve the aforementioned problems. The focus will be on the challenge of using as much information content as possible, by considering a population of cells and single cell information.

1.3 The Structure of Thesis

The thesis is organized as follows. Chapter 2 outlines related works on machine learning algorithms used in HCS, drug discovery, and cell biology. In chapter 3, we review the methodologies. The experimental design and results are described in chapter 4. Finally, a conclusion chapter is presented.

Chapter 2

Related Work

High Content Screening (HCS) technologies are used in cell biology, drug discovery, and biotechnology to recognize small molecules, RNAi, peptides, etc. [27, 65]. Recent developments in automated fluorescence microscopy systems have opened more doors to extract valuable phenotypic information from high content imaging screens [11, 20, 46]. The combination of automated fluorescence microscopy and high throughput technologies has led to experiments designed to infer relationships between genetic perturbations and cellular phenotypes.

In genomics screens, researchers tend to discover relations between genes by imaging cells with different genetic perturbations such as RNA interference or a genetic deletion [12, 32, 57]. High Content Screening of drug candidates using in vitro cell assays has led to more informed drug discovery in pharmaceutical research [42, 51]. Research on drug response screens in mammalian cells [47, 49, 53], protein localization in yeast [9, 17, 66], and cell morphology characterization [69, 70] are examples that utilized these technologies.

One of the promising works that highlights using imaging in comparison to lower dimensional features was done by Vizeacoumar et al. [67] that combines High Content Screening with synthetic genetic arrays to discover 122 additional genes involved in spindle function that were not recognized previously in a colony fitness based genetic interaction screen [68].

These High Content Analysis methods deal with screening thousands of cells which can provide a number of parameters for each cell such as nuclear size, nuclear morphology, DNA replication, etc. The success of these systems is dependent on automated image analysis. Analyzing this high throughput high-dimensional problem requires machine learning algorithms. Automatically classifying or profiling phenotypes, as was done by Chong et al. [17](2015), are preferable alternatives to manual assessment. Figure 2.1 shows a traditional work-flow for cellular classification. On the other hand, learning millions of cells from thousands of images requires a model with tremendous learning capacity.

In image-based profiling of cellular morphology, Ljosa et al. [47] compared five profiling algorithms including Means, Kolmogorov-Smirnov (KS) statistic, Normal Vector to Support-Vector Machine Hyperplanes, Gaussian Mixture Modeling, and Factor Analysis using the same experimental configuration, which is also tested on the BBBC021 dataset [10]. After segmenting these images, feature extraction was performed using CellProfiler software. 453 features in total were extracted per cell including size, intensity, shape, texture and neighborhood information. The profiles produced were assessed by classifying each treatment condition into its MOA. The term mechanism of action (MOA) refers to a sharing of similar phenotypic outcomes among various compound treatments, rather than referring strictly to modulation of

a particular target or target class [47]. For assigning each treatment to an MOA, a K-nearest neighbor classifier was applied. The Factor Analysis method was considered as the best performing profiling algorithm with a mean accuracy of 94% on just a small part of the dataset including 10% of all samples and not all the data [47]. Singh et al.[60] focused on improving accuracy by illumination correction in the images. Moreover, they realized that the Mean profiles are more robust than Factor Analysis profiles.

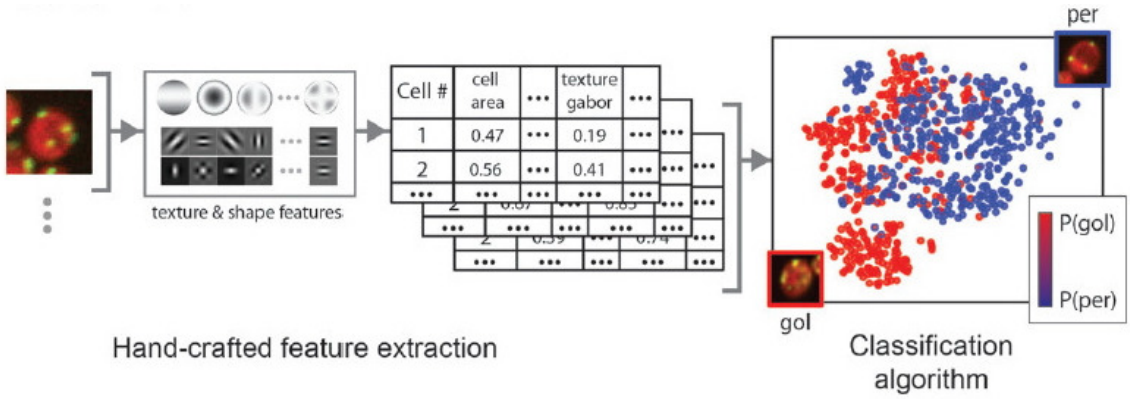

Figure 2.1: A usual work-flow for cellular classification [39]

Recently, deep learning algorithms have overcome object recognition issues. Figure 2.2 shows a deep learning work-flow for cellular classification. Deep learning has been applied to profiling cells for profile aggregation [72], MOA classification from CellProfiler features [36] and MOA classification from raw images [39]. Zamparo et al. used autoencoders for dimensionality reduction to improve the quality of profiles [72]. Kandaswamy et al. used the 453 features of each cell through CellProfiler software for a particular MOA class [36], similar to the work of Ljosa [47]. Kraus

et al. [39] tested their model on raw images in the BBBC021 dataset as applied in our study and got better results in comparison to traditional methods. This method locates regions with cells and adds a segmentation step to classify MOAs.

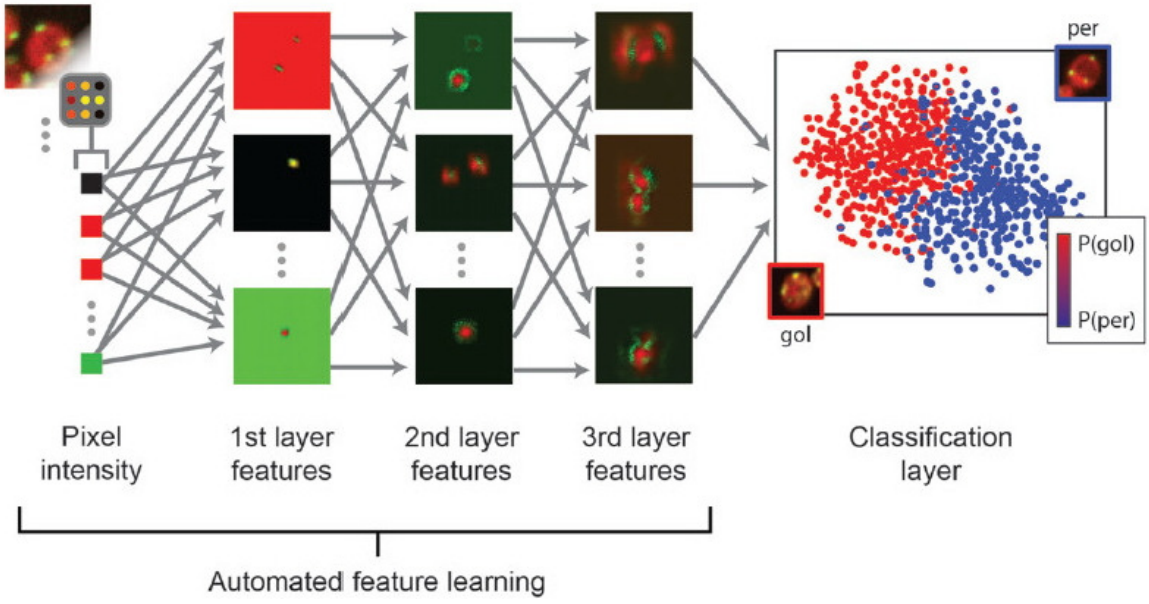

Figure 2.2: Deep Learning work-flow for cellular classification [39]

2.1 Machine Learning

Researchers have to utilize automatic algorithms to classify or cluster cells that they cannot evaluate manually [1] because HCS experiments deal with millions of cell images [17, 62]. The two classes of machine learning algorithms applied are supervised learning and unsupervised learning. In supervised learning a human expert interprets a subset of cells that map cell features to the appropriate output while unsupervised learning tries to extract patterns such as clusters of cells with similar phenotypes

from unlabeled cells.

2.1.1 Supervised Learning

Supervised learning methods typically try to learn a discriminative decision boundary through a subset of data tagged manually into various classes. These algorithms learn to categorize instances by improving their internal variables during the training step to minimize the error between their predictions and the target labels. During the testing step, another dataset of labeled data is applied to evaluate the ability of algorithms to generalize when encountering data not seen during the training step [6].

Research on protein localization [14, 17], drug profiling [47, 53], and changes in cell morphology [3, 70] are examples of phenotypes that have been trained through supervised learning algorithms. Support Vector Machines (SVM) [17, 24, 29, 49], random forests [37], neural networks [7, 31], and adaptive boosting [35] are supervised learning algorithms that have been applied to classify cellular phenotypes. Support Vector Machines attempt to find an optimal hyper plane that classify data points belonging to different categories [18]. Random forest [8] and adaptive boosting [22] are ensemble learning algorithms that synthesize multiple weak machine learning algorithms to provide a strong machine learning algorithm.

Neural networks are made of layers with artificial neurons that improve their parameters during the training step [55]. Because of recent success in training multi-layer neural networks, as deep learning algorithms, these algorithms currently reach innovative performance on many domains.

2.1.2 Unsupervised Learning

Sometimes a subset of labeled data is not accessible within a dataset, for example in HCS cases, when phenotype classes are not known in advance or are too subtle to accurately recognize manually. In such cases, unsupervised learning methods can be applied to learn patterns within this kind of dataset [26]. These methods tend to learn the structure of the dataset by finding clusters of data points that are similar. These strategies try to minimize a distance criterion between data points within the similar cluster and maximize the distance between data points belonging to various clusters [6].

One of the most important branches of unsupervised learning is clustering algorithms to include hierarchical clustering, K-means clustering, and Gaussian mixture models (GMM) that are applied to assemble unlabeled cells. Hierarchical clustering establishes a tree diagram by considering each data point as a cluster and unifying clusters recursively through a distance criterion [25]. Hierarchical clustering was applied to recognize distinct clusters of mRNA localization patterns in a large-scale fluorescence in situ hybridization (FISH) screen by Battich et al. [4]. Moreover, affinity propagation is one of the popular clustering algorithms that has been widely applied on gene expression [38, 45]. This method recognizes data points representative of clusters as perfect instances through looking at similarities amongst all pairs of data points [23]. K-means and Gaussian mixture models (GMMs) update repeatedly the cluster centers and the data points assigned to the nearest cluster center through the EM (expectation-maximization) algorithm [19].

In drug profiling, Ljosa et al. [47] evaluated various profiling algorithms on a small

portion of the BBBC21 drug screen dataset [10] and realized that Factor Analysis algorithm can achieve the best results. Young et al. [71] discovered that compounds that produce the same phenotypic profiles have the same chemical structures by the factor analysis method. PCA (Principal Component Analysis) was used to reduce the feature space and eventually GMMs were applied to measure cellular phenotype heterogeneity in cancer societies by Singh et al. [59]. Zhong et al. [73] evaluated unsupervised methods such as Hidden Markov Models and GMMs for finding cell cycle stages in human tissue culture cells indicating a fluorescent chromatin marker. Kurita et al. [41] performed a combination of profiles extracted from High Content Screening and untargeted metabolomics analysis to recognize modes of action of natural product extracts.

Another useful application of unsupervised algorithms is visualizing or exploring high dimensional datasets.

Chapter 3

Methodology

This thesis focuses on the challenge of using information content that is as high as possible, by considering cell populations and per-cell information to provide a system for High Content Analysis.

Recently, deep learning methods have been applied with success in a variety of applications [30, 5, 21, 54, 15, 16]. Moreover, deep learning algorithms can be fed with raw data and find important representations automatically for detection or classification. It has proven to be a very helpful tool for discovering complicated structures in high-dimensional data and thus is helpful for various fields of science and business.

In this work, we developed five deep learning algorithms to predict MOAs based on phenotypic screens of a population of cells in each image and single cells separately. We demonstrated that our models can predict the MOA for a compendium of drugs that modify cells without any segmentation and feature extraction steps. Previous works have only used a small subset (15%) of the BBBC021 dataset as training and test datasets. We believe that using this small volume dataset can lead to overfitting.

Most of Object recognition tasks generally involve a single centered object per image and current deep learning algorithms have not been applied to images with multiple objects such as microscopy images with a number of cells. Moreover, cell images have different characteristics; while objects seem in a constant scale and position, they are noisy and dimly lit. In this work, we provide five models inspired from AlexNet [40], VGG [58], and GoogleNet [64] deep neural network algorithms without using segmentation and feature extraction modules for single and multiple object recognition in High Content Screening images as a case study. Moreover, the proposed models predict the mechanism of actions for a compendium of drugs that alter cells by using all features in DNA, F-actin, and B-tubulin channels. Furthermore, these models predict the MOAs without a need for single cell identification.

Most of the current methods use CellProfiler software [11] for preprocessing and segmenting cell images and rely on hand-tuned detection and feature extraction for each cells image [39]. Our models can predict the MOA for a compendium of drugs that alter cells without using CellProfiler software and any segmentation and feature extraction steps, which is a significant improvement of traditional methods. These models trained on a big High Content Screening dataset (MCF-7 breast cancer cells - BBBC021), available from the Broad Bioimage Benchmark Collection [48], to predict the mechanisms of action for a compendium of drugs and achieved the best results ever reported on this dataset. Moreover, we found some hidden and unknown aspects of this dataset. After visualizing the dataset through PCA and t-SNE algorithms, we realized this dataset is imbalanced. In order to balance the BBBC021 dataset, Random Undersampling, Random Oversampling, and SMOTE algorithms were used [13] .

We demonstrated that if the architecture and selection of parameters of the deep learning models are configured properly, we do not need to use multiple instance learning [39] for classifying cells or predicting MOAs. On the other hand, recent research indicates that deep learning algorithms can learn features directly from the data [28, 43, 63]. In this work, deep learning algorithms are applied to predict MOAs from cell images directly. These models enable faster MOA prediction than the classical methods with image segmentation and feature extraction. Moreover, they perform better without human handcrafted features in comparison to traditional methods.

Most of the profiling methods were evaluated on scales at the resolution of single cells. The previous works usually did not work on a population of cells in a sample and only worked on single cells separately in each sample. However, compound treatments usually influence most cells in a sample (population of cells) and thus works on single cells are inadequate to predict the MOAs in RNAi screens. Furthermore, these models predict the MOAs without the need for single cell identification. This work is presented as a strategy to evaluate the effect and safety of drug candidates in complex biological systems. Our models are focused on morphology because there is a wide diversity of subtle cellular responses in images as samples.

3.1 Visualizing Data

Any data related challenge starts by exploring data itself. Our dataset has a high number of dimensions. Therefore, visually exploring this dataset can help us see the distributions of variables, classes, and correlations between them. Our work focuses on applying two techniques for visualizing our data, namely, t-SNE (T-Distributed Stochastic Neighbouring Entities) and a combination of PCA and t-SNE.

3.1.1 PCA (Principal Component Analysis)

PCA uses the correlation between some dimensions and provides a minimum number of variables that keep the maximum amount of information [52, 34, 33, 56].

3.1.2 t-SNE(T-Distributed Stochastic Neighbouring Entities)

In this section, a short introduction to t-SNE [50] is described. t-SNE presents overall similarities between data points in the high-dimensional space as a symmetric joint probability distribution P. In addition a joint probability distribution Q is achieved that defines the similarity in the low-dimensional space. t-SNE tries to find the best representation Q in the low dimensional space that represents P. In order to find this representation the positions in the low-dimensional space are optimized to minimize the Kullback-Leibler(KL) divergence between the probability distribution P and Q as a cost function C.

$$C(P,Q) = KL(P||Q) = \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} p_{ij} ln(\frac{p_{ij}}{q_{ij}}) \tag{3.1}$$

Maaten and Hinton [50] describe the working of t-SNE as: "t-Distributed stochastic neighbor embedding (t-SNE) minimizes the divergence between two distributions: a distribution that measures pairwise similarities of the input objects and a distribution that measures pairwise similarities of the corresponding low-dimensional points in the embedding".

3.2 Balancing Data

In most of the public datasets that are available the sample classes were balanced. In other words, the number of samples of each class was approximately the same. Researchers usually use cleaned up datasets in order to focus on specific algorithms without considering other issues. Usually, samples are presented similar to Figure 3.1 in which points and colors of points indicate samples and their classes respectively.

The main goal of a supervised machine learning algorithm is to learn a separator to recognize the two classes. You can see some decision boundaries using a variety of machine learning algorithms as separators in Figure 3.2. Machine learning algorithms are usually less sensitive to discovering the minority class and more sensitive to the majority class thus our main goal is balancing the dataset before feeding them into a machine learning algorithm. In many cases, if we do not balance the dataset, the classifier output would be biased. When we visualize real datasets, one of the first points we see is that they are imbalanced and noisier in comparison to more frequently datasets. The distribution of real data usually looks like the Figure 3.3.

In real problems, datasets can be imbalanced more. Most fraud detection datasets are strongly imbalanced. For example, around 2% of credit card accounts are scammed

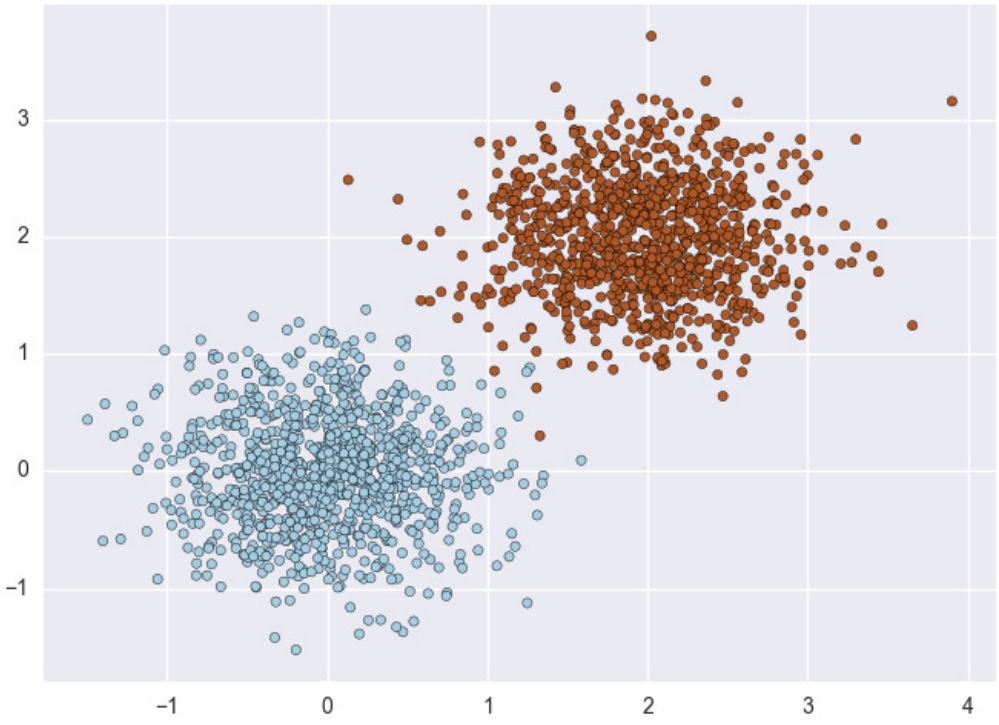

Figure 3.1: Balanced dataset.

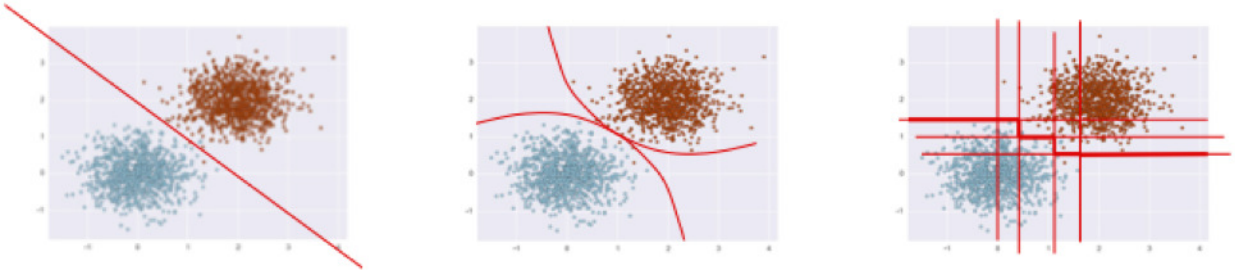

Figure 3.2: Using some machine learning algorithms as separators.

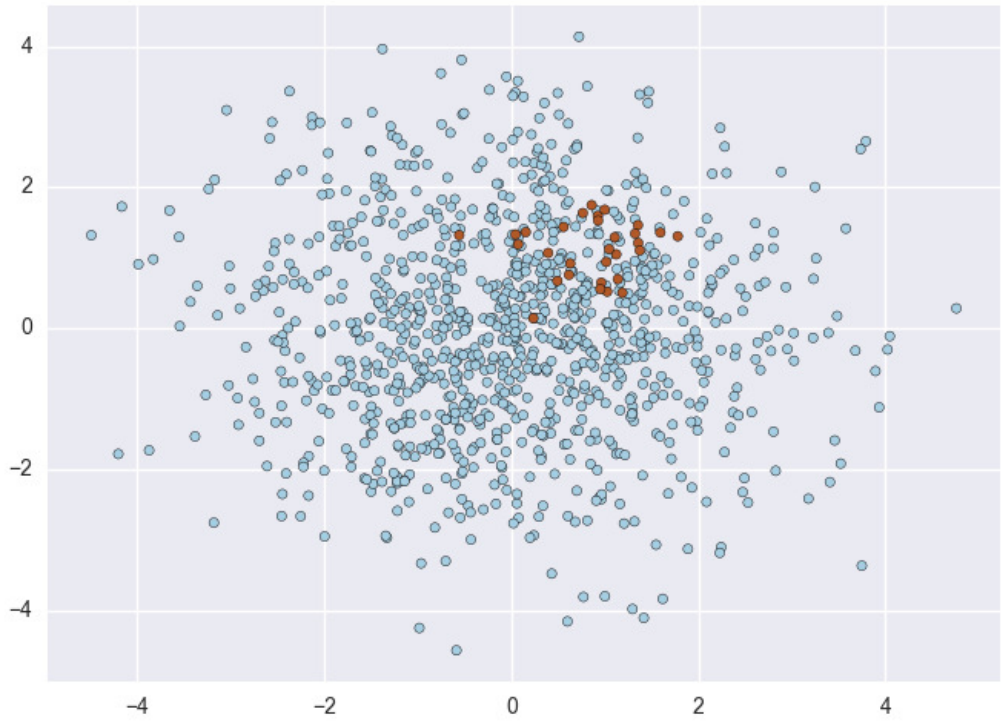

Figure 3.3: The classes are imbalanced.

each year. Fault rates for factory products usually are about 0.1%. In these types of problems, machine learning algorithms are applied to learn a large number of negative samples and a small number of positive samples. Traditional machine learning algorithms usually are biased when encountering such problems because their loss functions try to minimize error rate based on majority classes not minority classes. Machine learning algorithms ignore minority samples in the worst situations. There are some ways for balancing the dataset, such as oversampling the minority class, undersampling the majority class, synthesizing new minority classes, SMOTE algorithm, Removing minority samples, and switch to an Anomaly Detection frame.

3.2.1 Oversampling and Undersampling Methods

The easiest method for solving problems with imbalanced data is adjusting the dataset until it is balanced. The Random Undersampling method removes some samples from the majority class randomly. The random over sampling method replicates minority samples to increase the number of samples in the minority class [13] (Figure 3.4). Some machine learning scientists believe that Oversampling is better than Undersampling since it produces more data while Undersampling drops data. Repeating data decreases the variance of variables in the dataset.

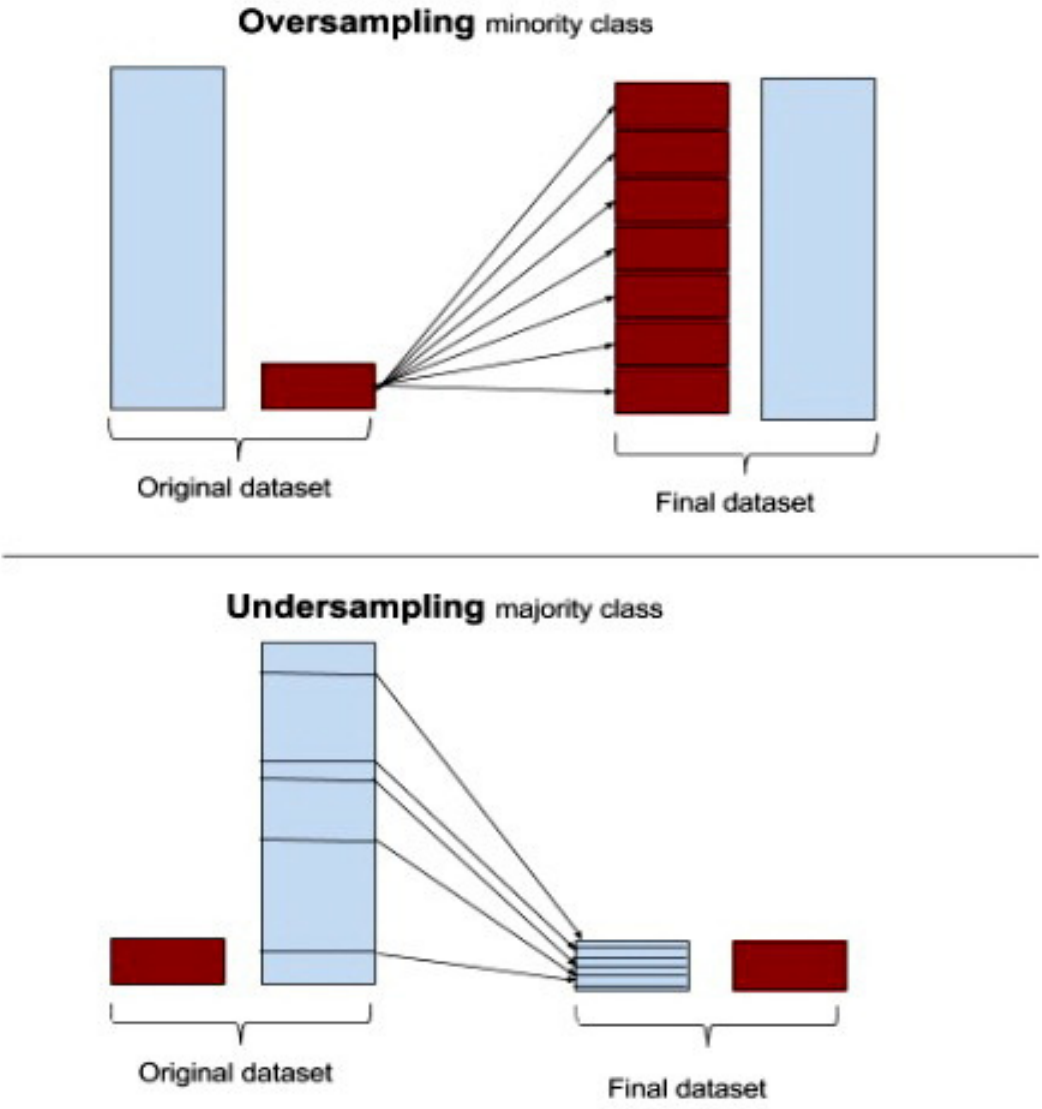

Figure 3.4: Oversampling and Undersampling Methods

3.2.2 SMOTE Algorithm

One of the best algorithms for Oversampling is the SMOTE (Synthetic Minority Over-sampling Technique) method that, instead of repeating samples, produces new samples through interpolations of the minority class and uses Undersampling on the majority class [13]. Because of all of the abovementioned reasons in section 3.2.1, we used the SMOTE algorithm that does not use re-sampling of samples but synthesizes new ones. This method produces new minority samples by interpolating between existing samples. Working by interpolating between minority samples is the main limitation of the SMOTE algorithm because can not produce new exterior regions of minority samples while it can only fill in the convex hull of minority samples in the dataset (Figure 3.5).

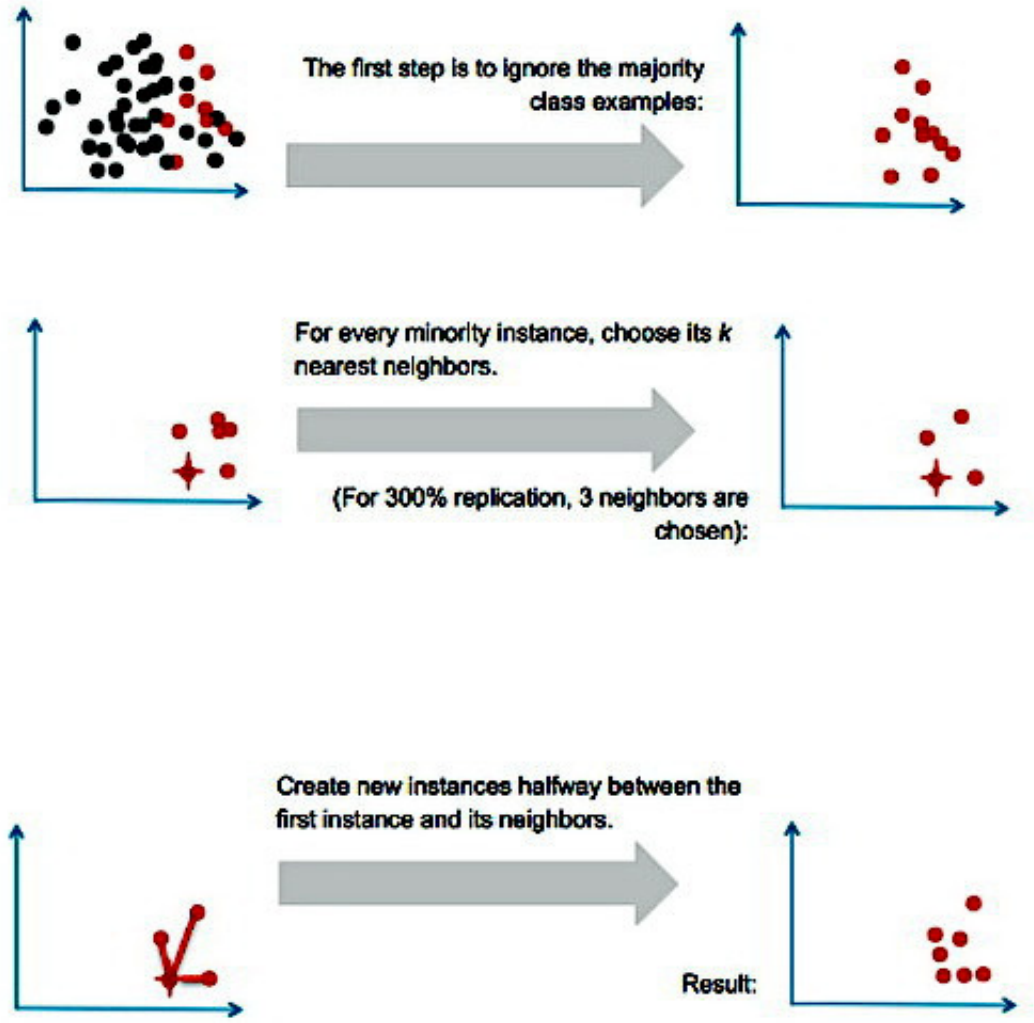

Figure 3.5: SMOTE Algorithm [13].

3.3 Models

3.3.1 A model inspired by AlexNet

Our first model was inspired from AlexNet[40]. AlexNet was the first deep learning model to win the Imagenet Challenge in 2012. The network has a very similar architecture as LeNet [44] by Yann LeCun but the number of weights is much greater and the shapes change from layer to layer because it is deeper with more filters per layer and different convolutional layers (11*11, 5*5, 3*3 convolutions). We added dropout in the input layer and two last convolutional layers before the final layer to avoid overfitting. We converted the three last fully connected layers in the original AlexNet to the convolutional layers before going to the output. The probability that activations are kept in the dropout in the input layer and two last convolutional layers before the final layer are supposed to be 0.8 and 0.5 respectively during training and 1.0 during testing. Our model has eight convolutional layers. There are 64 to 4096 filters within each convolutional layer and the filter size ranges from 1*1 to 11*11. A single image (from the DNA or F-actin, or B-tubulin stain channels) is used as an input in the input layer. ReLU activation functions are used after every convolutional layer. Max pooling of 3*3 with stride 2 is applied to the outputs of layers 1, 2 and 5. In order to decrease computation, a stride of 4 is considered at the input layer of the model. The original AlexNet used LRN in layers 1 and 2 before the max pooling. LRN is not being used anymore as a favorite component in other convolutional neural networks thus the LRN layers are removed in our model and the initializers were changed from random-normal-initializer to xavier-initializer. The basic architecture is shown in Figure 3.6. The number of filters in the first convolution is 64 and the

filter size 11*11 and the strides of the convolution along the height and width equals 4.

Figure 3.6: Our model inspired by AlexNet

3.3.2 Three models inspired by VGG

Our three other models were inspired from VGG-16 [58]. VGGNet was presented and developed at the ILSVRC 2014 competition by Simonyan and Zisserman. VGGNet, similar to AlexNet, only has 3*3 convolutions but lots of filters. Original VGG has two different models: VGG-16 (described here) and VGG-19. In this work we developed the modified versions of VGG, including VGG with 11 layers, VGG with 16 layers, and VGG with 19 layers.

VGG is deeper in comparison to the previous model (AlexNet based model). The larger filters (e.g. 5*5) are made from multiple smaller filters (e.g. 3*3), which have fewer weights to reach the identical receptive fields for balancing out the cost of moving deeper. Moreover, all the convolutional layers have the similar filter size of 3*3.

VGG is a modification of AlexNet that instead of having a lot of hyperparameters has a simpler network. It focuses on having only these blocks:

convolutional layer = 3 * 3 filter, stride = 1, same

Max-Pool = 2 * 2 , stride = 2

Our architectures are shown in Figure 3.7. VGG-16 has around 138 million parameters and most of the parameters are in the fully connected layers, which can be a bit challenging to handle. It uses a total memory of 96MB per image for only forward propagation. The VGG uses the most memory in the earlier layers. The number of filters increases from 64 to 128 to 256 to 512. 512 was repeated twice. Pooling is the only one component that is responsible for shrinking the dimensions. There is another version of VGGNet called VGG-19 which is a bigger version.

For decreasing the number of parameters we converted the fully connected layers to convolutional layers. VGG does not use many hyper parameters therefore it is a simpler model. It applies 3*3 filters with the stride of 1 in convolution layers and SAME padding in pooling layers 2*2 with the stride of 2. The weight decay parameter (L2 regularization coefficient) was supposed to be 0.0005.

VGG-11 (Inspired by VGG-16)	Inspired by VGG-16	Inspired by VGG-19
Input(224×224 grayscale image from F-actin or β-tubulin or DNA channels)		
Dropout on input layer		
conv1: number of filters=64, filter size=3×3	conv1: number of filters=64, filter size=3×3	conv1: number of filters=64, filter size=3×3
	conv2: number of filters=64, filter size=3×3	conv2: number of filters=64, filter size=3×3
maxpool1: filter size=2×2	maxpool2: filter size=2×2	maxpool2: filter size=2×2
conv2: number of filters=128, filter size=3×3	conv3: number of filters=128, filter size=3×3	conv3: number of filters=128, filter size=3×3
	conv4: number of filters=128, filter size=3×3	conv4: number of filters=128, filter size=3×3
maxpool2: filter size=2×2	maxpool4: filter size=2×2	maxpool4: filter size=2×2
conv3: number of filters=256, filter size=3×3	conv5: number of filters=256, filter size=3×3	conv5: number of filters=256, filter size=3×3
conv3: number of filters=256, filter size=3×3	conv6: number of filters=256, filter size=3×3	conv6: number of filters=256, filter size=3×3
	conv7: number of filters=256, filter size=3×3	conv7: number of filters=256, filter size=3×3
		conv8: number of filters=256, filter size=3×3
maxpool4: filter size=2×2	maxpool7: filter size=2×2	maxpool8: filter size=2×2
conv5: number of filters=512, filter size=3×3	conv8: number of filters=512, filter size=3×3	conv9: number of filters=512, filter size=3×3
conv5: number of filters=512, filter size=3×3	conv9: number of filters=512, filter size=3×3	conv10: number of filters=512, filter size=3×3
	conv10: number of filters=512, filter size=3×3	conv11: number of filters=512, filter size=3×3
		conv12: number of filters=512, filter size=3×3
maxpool6: filter size=2×2	maxpool10: filter size=2×2	maxpool12: filter size=2×2
conv7: number of filters=512, filter size=3×3	conv11: number of filters=512, filter size=3×3	conv13: number of filters=512, filter size=3×3
conv7: number of filters=512, filter size=3×3	conv12: number of filters=512, filter size=3×3	conv14: number of filters=512, filter size=3×3
	conv13: number of filters=512, filter size=3×3	conv15: number of filters=512, filter size=3×3
		conv16: number of filters=512, filter size=3×3
maxpool8: filter size=2×2	maxpool13: filter size=2×2	maxpool16: filter size=2×2
conv9: number of filters=4096, filter size=7×7	conv14: number of filters=4096, filter size=7×7	conv17: number of filters=4096, filter size=7×7
Dropout		
conv10: number of filters=4096, filter size=1×1	conv15: number of filters=4096, filter size=1×1	conv18: number of filters=4096, filter size=1×1
Dropout		
conv11: number of filters=12(number of classes), filter size=1×1	conv16: number of filters=12(number of classes), filter size=1×1	conv19: number of filters=12(number of classes), filter size=1×1
Flatten (We Flatten the output of previous layer while maintaining the batch size. A flattened tensor with shape [batch size, k].)		
output = soft-max		

Figure 3.7: Our new models inspired by original VGG-16: new VGG-11, new VGG-16, new VGG-19.

3.3.3 A model inspired by GoogleNet

GoogleNet (Inception V1) from Google was the winner of the ILSVRC 2014 competition[64]. GoogleNet is a deeper convolutional neural network with 22 layers (Figure 3.8). It has an Inception module that is made of parallel connections (Figure 3.9).

Each parallel connection uses various filters sizes (1*1, 3*3, 5*5) and 3*3 max-pooling and we concatenate their outputs eventually for the module output. GoogleNet has the impact of processing the input at different scales and then collect them so that the next level can get compressed features from various scales simultaneously because of using multiple filter sizes. 1*1 filters were used as bottleneck layers because of decreasing the number of channels and eventually the number of weights.

The 22 layers include three convolutional layers, 9 Inception layers (each of which is two convolutional layers), and one final convolutional layer (Figure 3.8). The output of the final layer was flattened while it produced a flattened tensor with shape [batch size, k]. After that, a Softmax function was applied.

The Inception modules outputs are constrained to change because these modules are stacked on top of each other thus their spatial concentration is decreased(Figure 3.9). The architecture of this model consists of stacked Inception modules with max-pooling layers with stride 2. Although the original paper did not use batch normalization we found it beneficial. We used batch normalization in our model but we did not show it in the diagram (Figure 3.8).

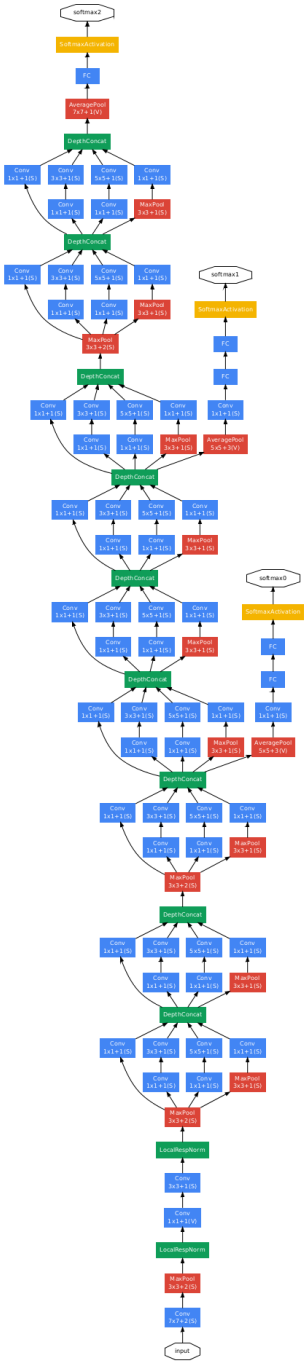

Figure 3.8: GoogleNet overall view [64]. Batch normalization was not shown in this diagram.

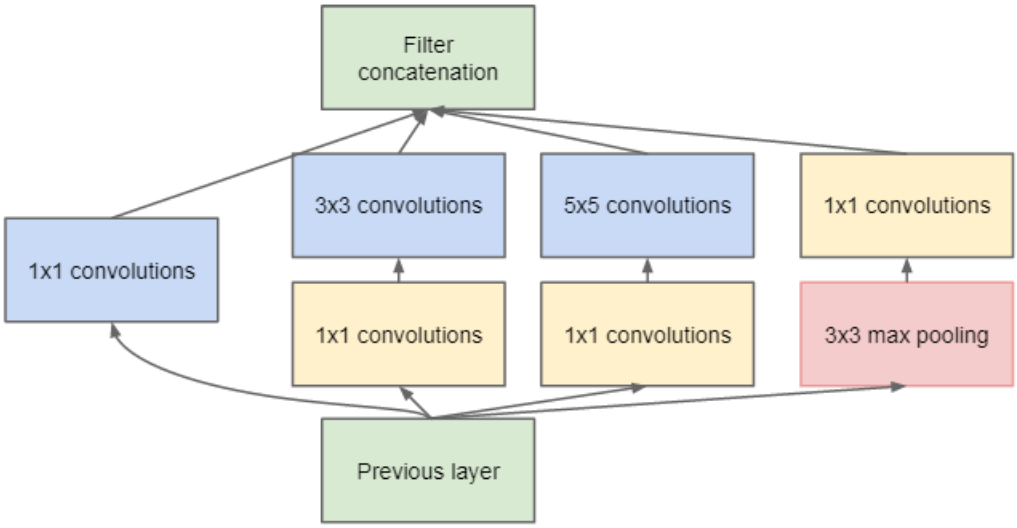

Figure 3.9: Inception module with bottleneck layers [64].

Chapter 4

Experimental design and results

Recent works on analyzing High Content Screening images are focused on using gained features from CellProfiler software. Our models can predict the MOA for a compendium of drugs that alter cells, without using CellProfiler software and any segmentation and feature extraction steps which would imply a significant improvement on traditional methods. These findings were tested on the most popular High Content Screening big dataset, BBBC021 [10]. Deep learning algorithms have usually been tested on high quality datasets such as Imagenet, Cifar, etc., and their performance on challenging datasets, including low quality samples, is rarely tested (e.g. High Content Screening images with low quality). In this work we tested our models on a High Content Screening breast cancer big dataset (MFC-7) as a case study. This study tests the ability of deep learning algorithms on samples with very low qualities. It is shown that deep learning algorithms can produce incredible results for predicting MOAs on low quality images without using any segmentation and feature extraction phases if their architectures and parameters are configured appropriately.

In this work, we investigate the effect of deep learning algorithms on the prediction of MOAs. We proposed five models for the prediction of MOAs from the MFC-7 (BBBC021) dataset into 12 MOAs. All previous works focused on single cell identification. We believe that single cell recognition is not the most useful for predicting MOAs because compound treatments usually impact most cells in a sample in a variety of ways, not just a single cell. Therefore, we worked on a population of cells in samples to predict MOAs in RNAi screens. We believe that not only can our models predict a specific phenotype through a population of cells in a particular sample but they also can predict specific phenomenon by the low proportion of cells in a sample. We think there are some relations between cells in a channel image that have not been considered. Object recognition tasks generally have a single centered object per image. Current deep learning algorithms have not been applied to images that include multiple specific complex objects such as microscopy images because there are a number of objects (cells) in these images.

4.1 Data

We used dataset BBBC021 [10] concerning molecular cancer therapeutics (MCF-7 wild-type P53 breast cancer cells) available from the Broad Bioimage Benchmark Collection [48]. A comprehensive study has not been performed on the BBBC021 dataset. The images were treated for 24 h with a collection of 113 molecules at eight concentrations. The ground-truth data set includes the mechanisms of action of 103 compound-concentrations (38 compounds at 1-7 concentrations each) [48]. For all 103 treatments, some compounds were active at up to seven concentrations, and some at

only one concentration, covering 12 mechanisms of action classes (Figure 4.1). All 103 treatments were classified into MOAs by assigning to the predicted class of each its most similar MOA. The cells were imaged by fluorescent microscopy, and labeled for DNA, F-actin, and B-tubulin. For example, DNA, F-actin, and B-tubulin channels for DNA damage MOA are shown in Figures 4.2, 4.3, and 4.4, where the compound is mitomycin C and the concentration is 0.3 at week 4. There are 39600 image files, (13200 views in three channels), in TIFF format (16 bit). All the images have been converted to grayscale images (8 bit images). The original images are all 1280 by 1024 pixels. These images were converted to 224 by 224 pixels grayscale images to feed our machine learning models. Therefore each image has 224*224 dimensions, and each one holds the gray level of one specific pixel.

On deep learning projects, usually the first crucial step is to engage with a large amount of data or a large number of images (e.g. a couple of million images in ImageNet). In such a case, it is inefficient to load every single image from the hard disk separately, use image preprocessing, and pass the image to the deep learning algorithm to train or test. Preprocessing requires time, and reading multiple images from a hard disk is more time consuming than having them all in a single file and reading them as a single bunch of data. Luckily, there are different useful data models and libraries, and one that we are interested in using for this project is called HDF5. In this work, we saved a large number of images in a single HDF5 file and then loaded them from the file as a single batch. It is not important how large the dataset, nor whether it is larger than our memory size or not. HDF5 produces tools to manage, manipulate, compress, and save the data.

The dataset includes 39600 images, including 13200 field-of-view images in three

channels (DNA, F-actin, and B-tubulin channels). This is a large dataset, but some samples are unlabeled and some are artificial (DMSO samples). The dominance of positive bias in the mock-treatments (DMSO) suggests that the accuracy of the true treatments may also be overestimated. Therefore, the unlabeled and DMSO samples were removed from the dataset. The number of samples for all classes after removing unlabeled and DMSO samples is 7584. We divided this dataset into two parts: first, 6444 samples of DNA, B-tubulin, and F-actin channels as training data, and second, 1140 samples of DNA, B-tubulin, and F-actin channels as testing data. Each image in our dataset (DNA, actin filaments, or B-tubulin) has a corresponding label, a number between 0 and 11, representing the MOA (Table 4.1).

Mechanism of action	Compound	Concentrations [μM]
Actin disruptors	cytochalasin B	10.0, 30.0
	cytochalasin D	0.3
	latrunculin B	1.0, 3.0
Aurora kinase inhibitors	AZ-A	0.1, 0.3, 1.0, 3.0, 10.0, 30.0
	AZ258	0.1, 0.3, 1.0
	AZ841	0.1, 0.3, 1.0
Cholesterol-lowering	mevinolin/lovastatin	1.5, 5.0, 15.0
	simvastatin	2.0, 6.0, 20.0
DNA damage	chlorambucil	10.0
	cisplatin	10.0
	etoposide	1.0, 3.0, 10.0
	mitomycin C	0.1, 0.3, 1.0, 3.0
DNA replication	camptothecin	0.003, 0.01, 0.03
	floxuridine	10.0, 30.0
	methotrexate	10.0
	mitoxantrone	0.003, 0.01
Eg5 inhibitors	AZ-C	0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1.0
	AZ138	0.03, 0.1, 0.3, 1.0, 3.0
Epithelial	AZ-J	1.0, 3.0, 10.0
	AZ-U	1.0, 3.0, 10.0
	PP-2	3.0, 10.0
Kinase inhibitors	PD-169316	3.0, 10.0
	alsterpaullone	1.0, 3.0
	bryostatin	0.3
Microtubule destabilizers	colchicine	0.03
	demecolcine	0.3, 1.0, 3.0, 10.0
	nocodazole	1.0, 3.0
	vincristine	0.003, 0.01, 0.03, 0.1, 0.3, 1.0, 3.0
Microtubule stabilizers	docetaxel	0.03, 0.1, 0.3
	epothilone B	0.1, 0.3, 1.0
	taxol	0.3, 1.0, 3.0
Protein degradation	ALLN	3.0, 100.0
	MG-132	0.1, 3.0
	lactacystin	10.0
	proteasome inhibitor I	0.1, 3.0
Protein synthesis	anisomycin	0.3, 1.0
	cyclohexamide	5.0, 15.0, 50.0
	emetine	0.1, 0.3, 1.0

Figure 4.1: The ground-truth set [10].

Table 4.1: Mechanism of actions

Mechanism of action	Code of class
Actin disruptors	0
Aurora kinase inhibitors	1
Cholesterol-lowering	2
DNA damage	3
DNA replication	4
Eg5 inhibitors	5
Epithelial	6
Kinase inhibitors	7
Microtubule destabilizers	8
Microtubule stabilizers	9
Protein degradation	10
Protein synthesis	11

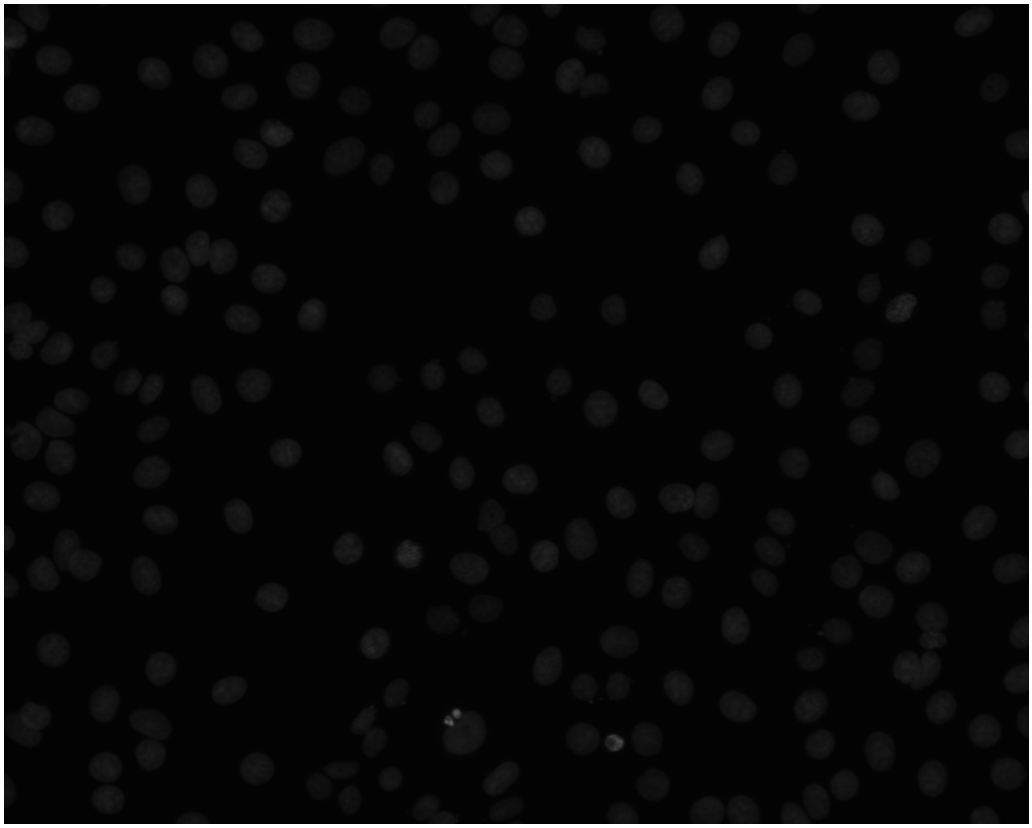

Figure 4.2: DNA channel image for DNA damage MOA when the compound is mitomycin C and concentration is 0.3 in week 4 [10].

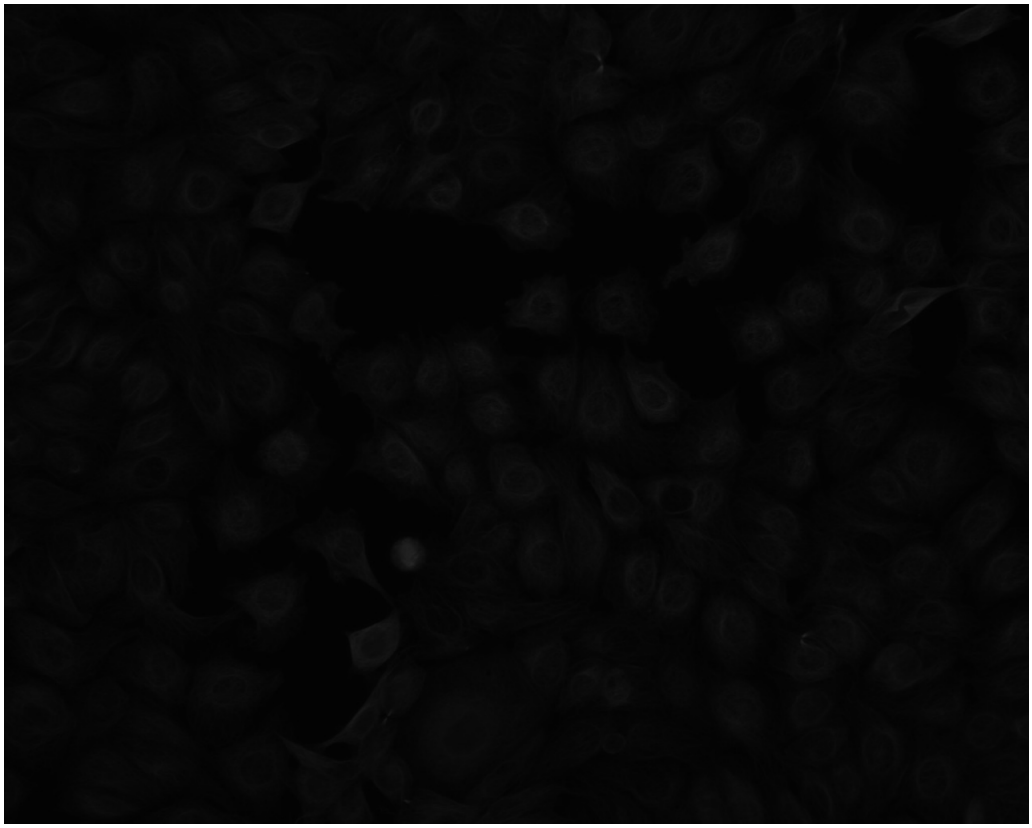

Figure 4.3: B-Tubulin channel image for DNA damage MOA when the compound is mitomycin C and concentration is 0.3 in week 4 [10].

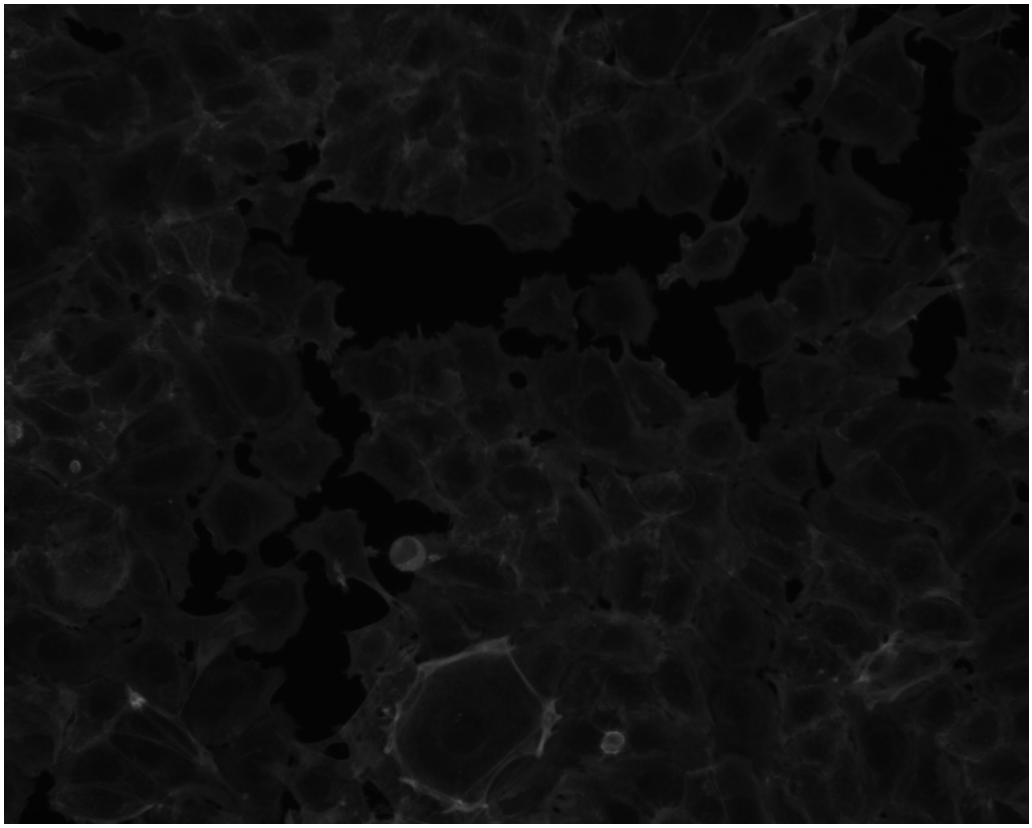

Figure 4.4: Actin channel image for DNA damage MOA when the compound is mitomycin C and concentration is 0.3 in week 4 [10].

4.2 Visualizing the Data

Any data related challenge starts by exploring data itself. The BBBC021 dataset has a large number of variables and a high number of dimensions. Therefore, visually exploring this dataset can help us see the distributions of variables, classes, and correlations between them. In this work, we focus on applying two techniques for visualizing the data. We visualized the original imbalanced data and balanced data

through two methods, namely, t-SNE and a combination of PCA and t-SNE. We want to reduce the number of dimensions for visualizing the dataset while trying to keep as much of the variation in the data as possible.

4.2.1 PCA

The PCA method is used for dimensionality reduction in the dataset while keeping as much of the information as possible regarding how the data is distributed.

4.2.2 t-SNE

One of the most popular methods for dimensionality reduction and visualizing high dimensional datasets is t-SNE (t-Distributed Stochastic Neighbor Embedding). In comparison to PCA, this method is a probabilistic algorithm. This method looks at the original data as an input to the algorithm and finds the representation of the data with fewer dimensions using matching both distributions (original data and lower-dimensional data). Moreover, this method compares the original data to the lower-dimensional data and then finds the closest approximation of the lower-dimensional data to the original data. The serious disadvantage of this method is that it is computationally laborious. In cases of very high dimensional data, this limitation may be addressed by using another dimensionality reduction algorithm before applying t-SNE. For example, PCA can be used for dense data or Truncated SVD can be used for sparse data. There is a significant improvement in comparison to the t-SNE visualization we used previously, without PCA. As you can see in figure 4.6, the High Content Screening samples are very clearly clustered in their own class. If we then

use a clustering algorithm to identify the separate clusters, we could probably assign new points to a label with a high degree of accuracy.

On the other hand, according to the notes before, the number of dimensions was reduced by PCA before feeding the data into the t-SNE algorithm. Subjecting raw data to the PCA algorithm produced a new dataset with 40 dimensions, and this was fed into the t-SNE algorithm (Figure 4.6). A scatter-plot of the first to 40th principal components that fed to t-SNE is created and the different types of classes (MOAs) identified by colour are shown in Figure 4.6. It is obvious that, some types of MOAs are clustered together in groups which means that the first to fortieth principal components and t-SNE are enough to visualize the specific types of MOAs.

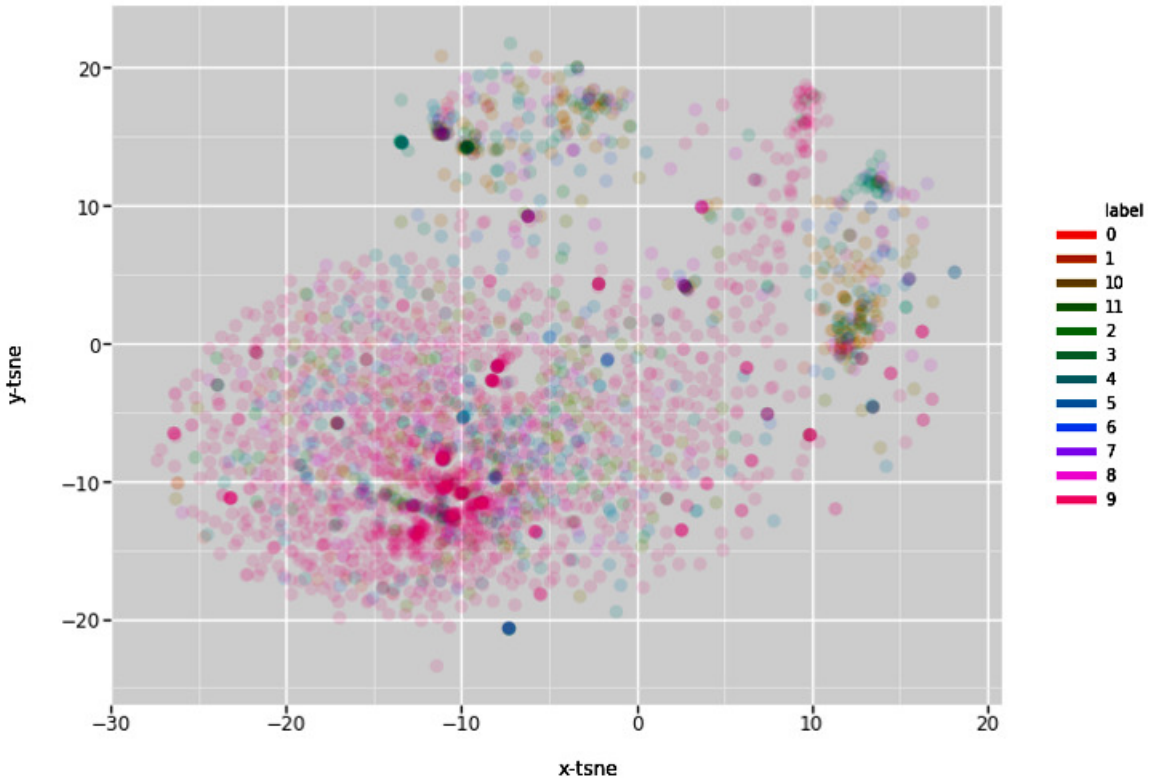

Figure 4.5: The original dataset was visualized through t-SNE algorithm. The dimensions colored by the code of classes(MOAs).

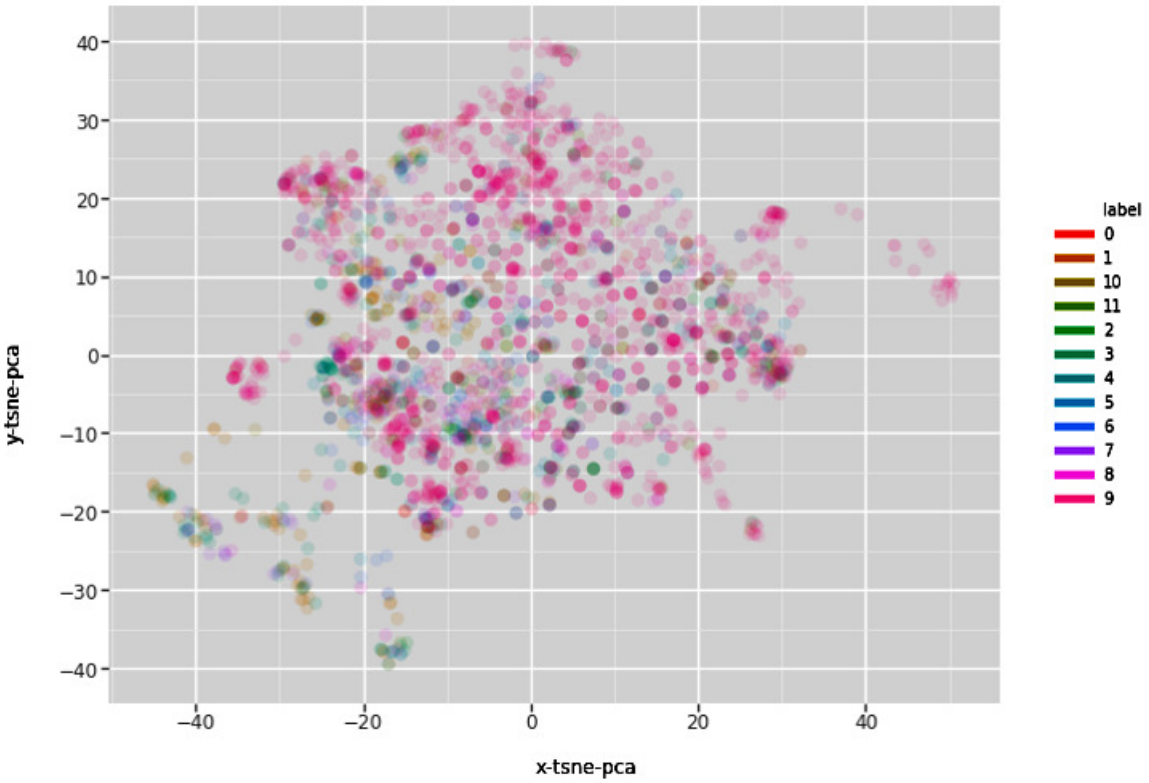

Figure 4.6: The original dataset was visualized through PCA + t-SNE algorithms. The dimensions colored by the code of classes(MOAs).

4.3 Producing Balanced Dataset

After visualizing and looking at the dataset carefully, it is obvious that the BBBC021 labeled dataset is highly unbalanced (Figures 4.5 and 4.6). Visualizing the data shows that there are a majority of samples in Microtubule stabilizers class (class 9) in comparison to the number of samples for other classes. We used three algorithms to balance the dataset, namely, Undersampling, Oversampling, and SMOTE [13]. Researchers have not considered the impact of imbalanced data on deep learn-

ing algorithms. Therefore, in this work, we investigate the impact of data on the performance of deep learning algorithms. A balanced dataset produced by the Undersampling method was visualized through the t-SNE (Figure 4.7) and PCA+t-SNE (Figure 4.8) algorithms.

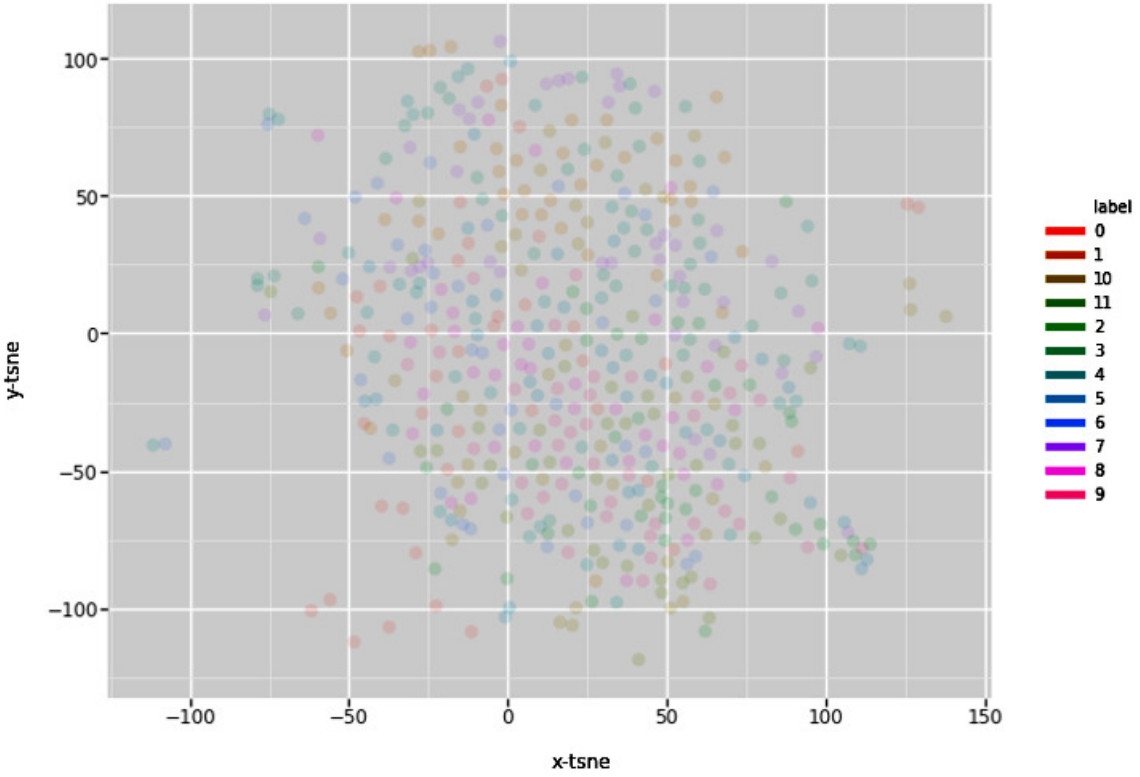

Figure 4.7: The balanced dataset was visualized through t-SNE after balancing the data by Undersampling. The dimensions are colored according to their class code (MOA).

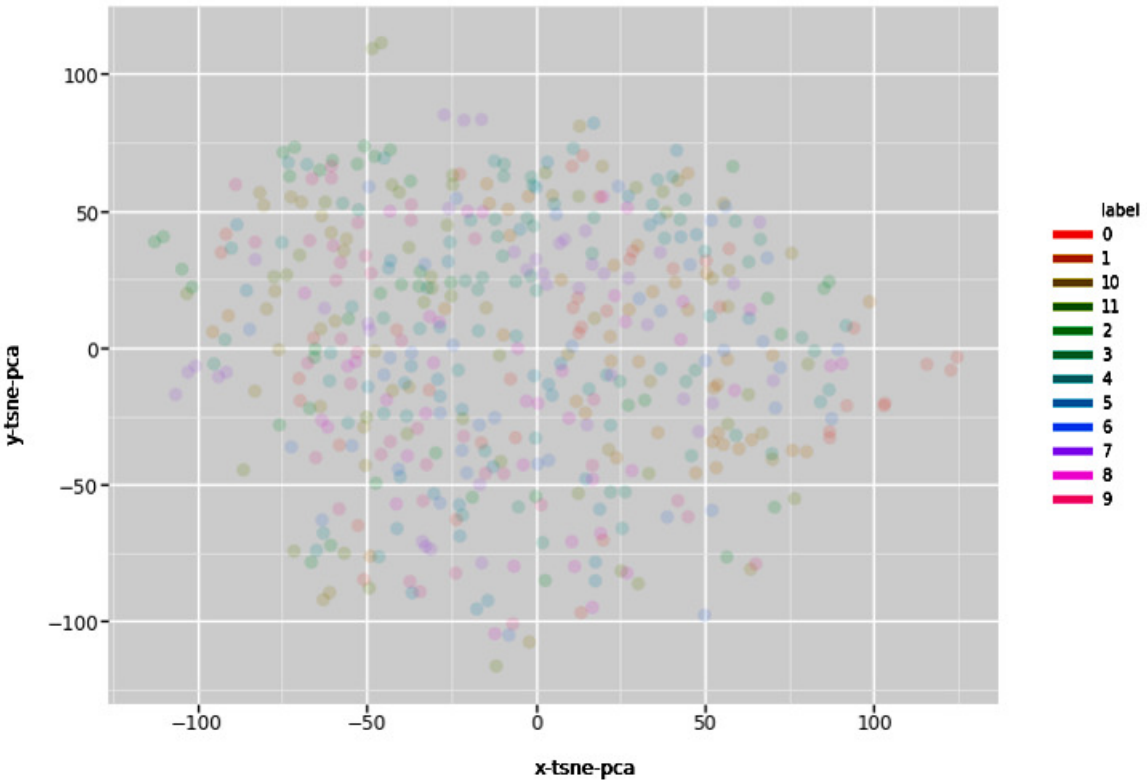

Figure 4.8: The balanced dataset was visualized through PCA + t-SNE algorithms after balancing data by Undersampling method. The dimensions colored by the code of classes (MOAs).

After using the Undersampling method as a balancing the data method, the number of samples is 1440. We selected 85% of the samples (images) from 103 treatments to train (1224 samples with one of the three channels namely, F-actin, B-tubulin, and DNA) and 15% (216 samples) to test the models.From Figures 4.7 and 4.8, it is clear that all the samples are positioned apart nicely and grouped together with their specific class. The number of samples after using the Oversampling SMOTE method is 51408 , 85% and 15% of these samples were taken from the training and testing

datasets, respectively. Concerning the Oversampling method, we reused samples in the minority class repeatedly, and then we feed them into our models. At this step, there will be replication. Thus, the train and test sets include the same samples. This can cause overfitting. They are made and shuffled by considering preserving the percentage of samples for each class.

4.4 Models Testing

The proposed models require a fixed input of dimensionality. Thus, the images were down-sampled to a constant resolution of 224*224 pixels. We only subtracted the mean and divided by the standard deviation of each channel in our training set as an image pre-processing step. Our models were trained on the raw gray level values of the pixels from the DNA, F-actin, and B-tubulin channels for each mechanism of action.

According to the results, our models can predict specific phenotypes through the population of cells in a particular sample. Our best model (inspired by GoogleNet and using SMOTE algorithm) achieved 86% accuracy on all samples with a population of cells in the large BBBC021 dataset without requiring segmentation or a feature extraction step. Moreover, in comparison to the previous works, our model reached 98% accuracy on a single cell for predicting MOAs. Our results show that five models, inspired by GoogleNet, AlexNet, and VGG are capable of recognizing multiple objects (cells) in images. These algorithms were tested three times and an accuracy average over these three replications was calculated. We determined the accuracy, top-2 error, top-3 error, confusion matrix, and normalized confusion matrix as objective

measures for estimating discriminatory ability. Moreover, our models can classify those MOAs that are difficult to classify with the human visual system. With SMOTE and GoogleNet, the prediction accuracy was 86% which is significantly better than any of the other models that were tested (table 4.6).

Keskar et al. [38] points out that having a very large batch size can reduce the generalization ability of the machine learning algorithm thus we supposed the batch size 16. Our models were tested on full images. We did not use any preprocessing process except mean subtraction and division by the standard deviation. The full images were resized to feed the models. We suppose each grayscale stain image is related to one of the 12 MOAs. Moreover, each sample can be relevant to one of the three channels including DNA, F-actin, and B-tubulin.

We trained our models with Stochastic Gradient Descent for 500 epochs with an initial learning rate of 0.001. At first, we used a fixed learning rate (0.001) for all layers, which we adjusted manually throughout training. Then Steepest Gradient Descent optimizer was used with adaptive learning rate. The results with adaptive learning rate show improvement in comparison to fixed learning rate during the training phase (Figures 4.9 and 4.10). We used the adaptive learning rate method as follows.

- `The initial learning rate is 0.001`
- `decay` = `learning rate` / `total steps`
- `learning rate` = `learning rate` $*1/(1 + decay * i)$.

As shown in Figures 4.30, 4.32, 4.34, 4.36, 4.38, more than 95% of the error on predicting MOAs comes from protein degradation and protein synthesis classes

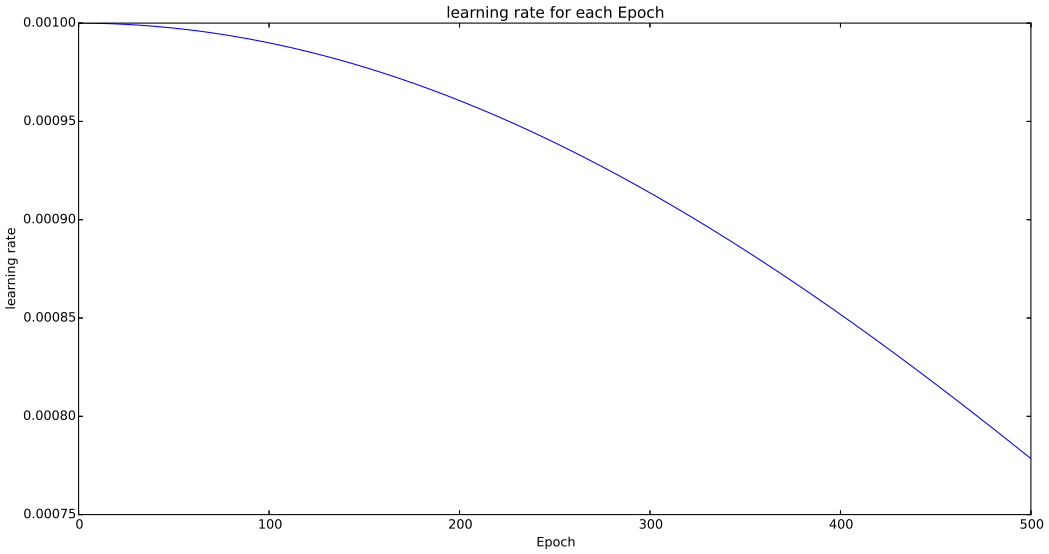

Figure 4.9: Adaptive learning rate

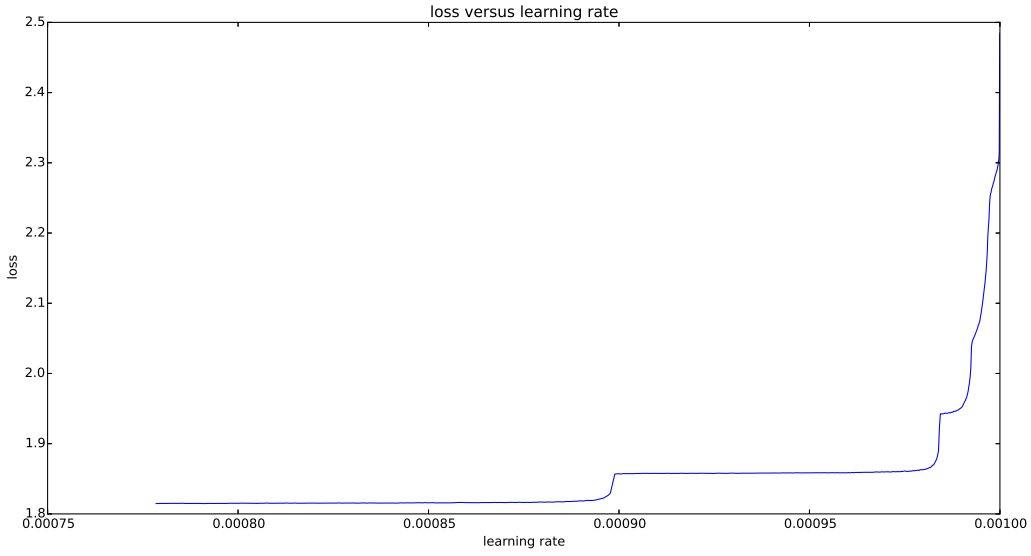

Figure 4.10: Adaptive learning rate and loss function

(SMOTE + our deep learning models is section 4.4.2). Average accuracy on all of the batches for each epoch was obtained as reported in Figure 4.25. Predicted accuracy was obtained by taking the average over three experimental replicates. Our VGG-11, VGG-16, GoogleNet, AlexNet models combined with SMOTE algorithm achieved 100% accuracy for predicting Actin disruptors, Aurora kinase inhibitors, Cholesterol-lowering, DNA damage, DNA replication, Eg5 inhibitors, Epithelial, Kinase inhibitors, Microtubule stabilizer, and Protein synthesis MOAs (normalized confusion matrices in Figures 4.30, 4.32, 4.34, 4.36). It is notable that our models mainly obtain their discriminatory capability from a combination of image features. The top-3 and top-2 errors also achieved to describe the accuracy of our algorithms (Figures 4.13, 4.14, 4.27, 4.28, 4.41, 4.42).

The similarity level of MOAs are shown in normalized confusion matrices in sections 4.4.1, 4.4.2, and 4.4.3. The confusion matrix is applied to evaluate the quality of the output of our models. The confusion matrix with and without normalization are shown in Figures in sections 4.4.1, 4.4.2, and 4.4.3. The diagonal elements represent the number of samples that predicted correctly in each class.

The under-sampled dataset could not provide satisfactory results because of the low number of samples while the results for combining over-sampling SMOTE method and deep learning algorithms are very satisfactory because it deals with many samples. Moreover, the sensitivity of our models will be increased by Undersampling. As shown in section 4.4.3 the results are very poor because our models were trained using a few samples. On the other hand, using the Oversampling method before feeding the data into classifier can produce overfitting problems.

Moreover, SMOTE algorithm effectively tries to solve the generalization problem

for minority class. Results are very satisfactory now. Especially for GoogleNet, we obtained accuracy 86% with the SMOTE algorithm (on all samples in the dataset). By using SMOTE algorithm before our models, we obtained almost perfect accuracy.

As expected, more data solved the problem regardless of using SMOTE algorithm as a smart Oversampling method. According to the results, SMOTE Oversampling algorithm properly is much better than Undersampling, for this dataset.

Our GoogleNet, AlexNet, and VGG-11 models achieved very good top-3 error (0.03) for all 12 MOA classes. Moreover, our VGG-16 and VGG-19 models reached 0.04 and 0.05 top-3 error respectively. Combination of SMOTE and our deep learning models take 80 hours to train on two Nvidia GPUs with 16 cores and 128 Gigabytes memory.

4.4.1 Our Deep Learning algorithms on imbalanced data

Table 4.2 shows the number of samples in the imbalanced raw dataset, the train, and test datasets. According to the results, many protein degradation and protein synthesis are misclassified as microtubule stabilizers MOA when imbalanced data and our deep learning algorithms including GoogleNet or VGG-11 are combined.

Many MOAs are misclassified as microtubule stabilizers when our models including AlexNet, VGG-16, and VGG-19 are applied on imbalanced data directly because the number of microtubule stabilizers samples is high in comparison to the other MOAs in our dataset.

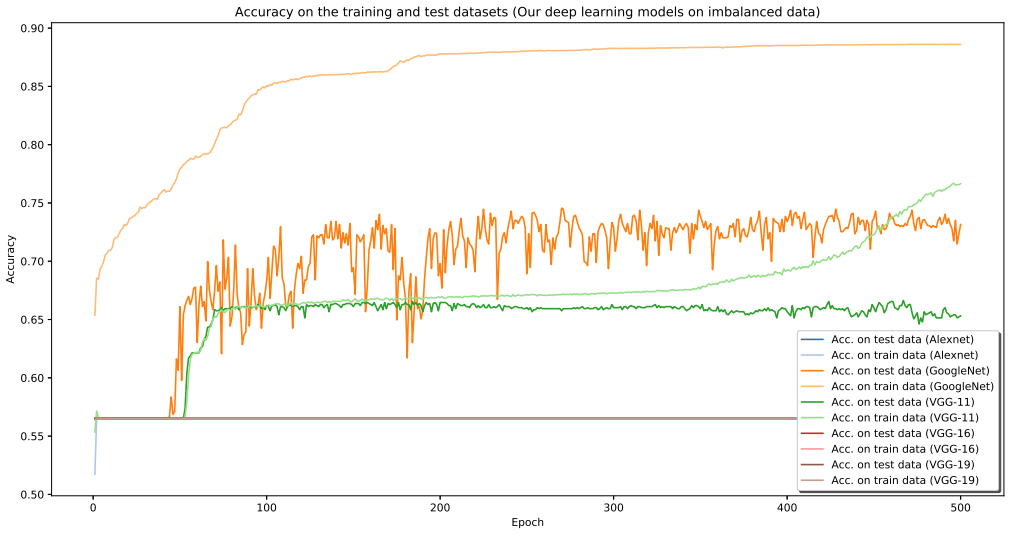

Figure 4.11: Accuracy on the training and test datasets (our deep learning models on the imbalanced data)

Table 4.2: The number of samples in the imbalanced raw dataset is 7584. The number of samples in the train dataset is 6446. The number of samples in the test dataset is 1138.

MOA (raw dataset)	Code of class	Number of samples	
		train dataset	test dataset
Actin disruptors	0	153	27
Aurora kinase inhibitors	1	367	65
Cholesterol-lowering	2	184	32
DNA damage	3	275	49
DNA replication	4	245	43
Eg5 inhibitors	5	367	65
Epithelial	6	225	39
Kinase inhibitors	7	102	18
Microtubule destabilizers	8	428	76
Microtubule stabilizers	9	3641	643
Protein degradation	10	214	38
Protein synthesis	11	245	43

Table 4.3: The accuracy on the imbalanced test dataset.

	Our models inspired by				
	GoogleNet	AlexNet	VGG-16 (VGG-11)	VGG-16	VGG-19
Accuracy	74%	56%	66%	56%	56%

Table 4.4: The top-2 and top-3 errors on the imbalanced test dataset.

	Our models inspired by				
	GoogleNet	AlexNet	VGG-16 (VGG-11)	VGG-16	VGG-19
Top-2 error	0.14	0.37	0.27	0.36	0.37
Top-3 error	0.07	0.29	0.21	0.31	0.29

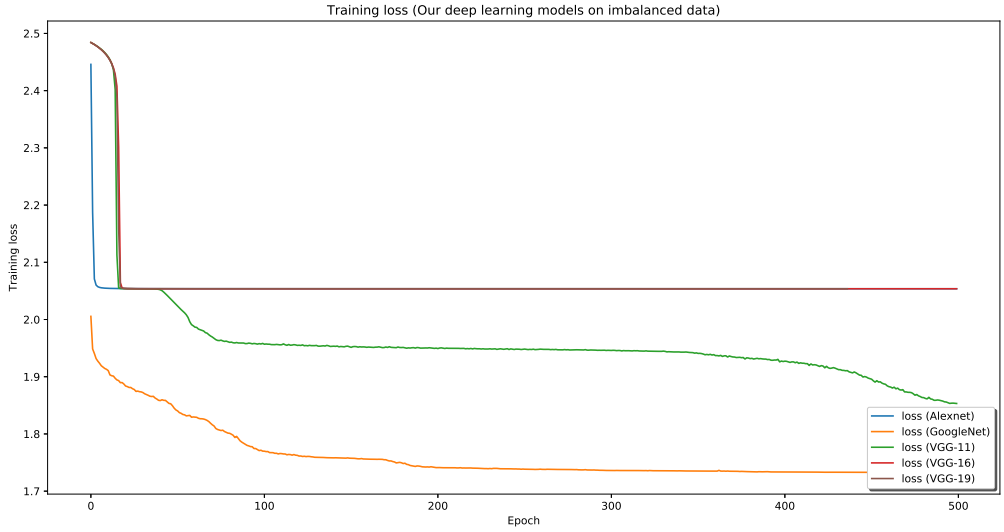

Figure 4.12: Training loss (our deep learning models on the imbalanced data)

Figure 4.13: top-3 error on the imbalanced dataset

Figure 4.14: top-2 error on the imbalanced dataset

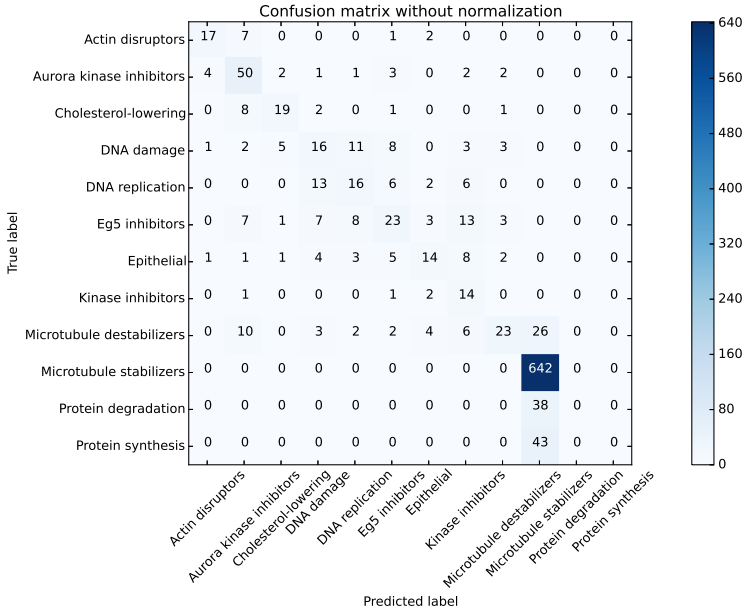

Figure 4.15: confusion matrix, our GoogleNet, imbalanced dataset.

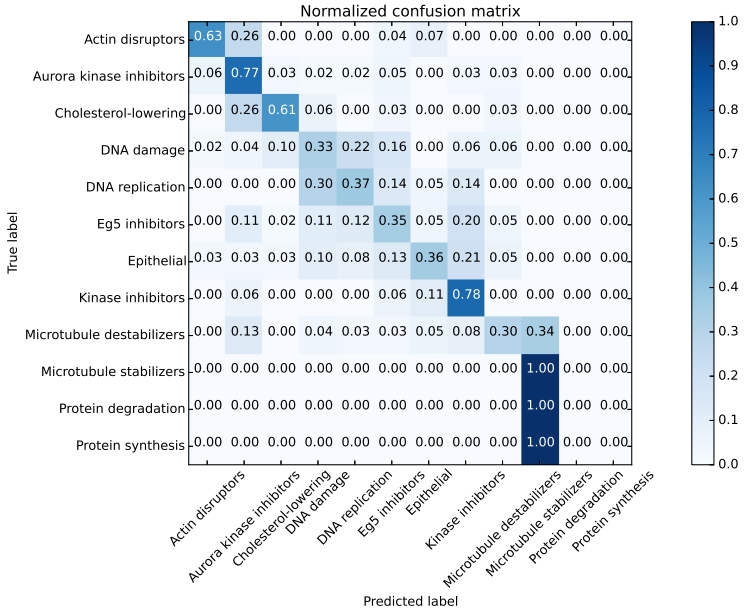

Figure 4.16: normalized confusion matrix, our GoogleNet, imbalanced dataset.

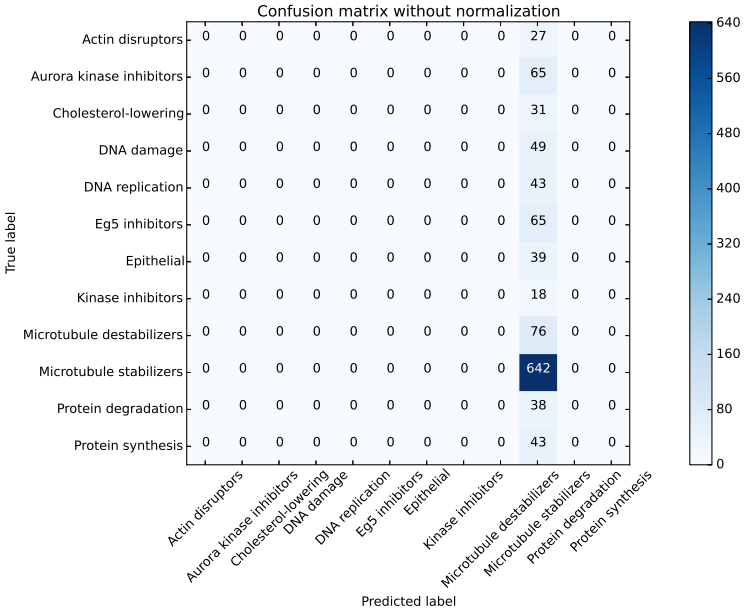

Figure 4.17: confusion matrix, our AlexNet, imbalanced dataset

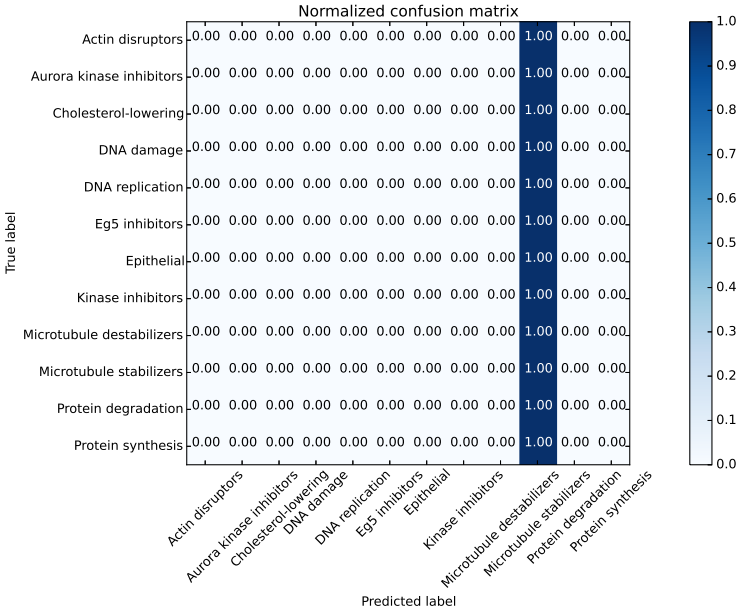

Figure 4.18: normalized confusion matrix, our AlexNet, imbalanced dataset.

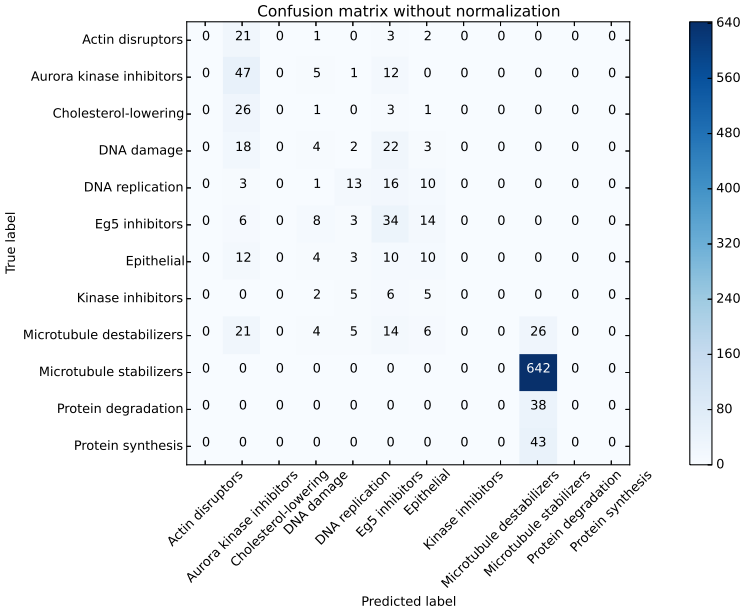

Figure 4.19: confusion matrix, our VGG-11, imbalanced dataset.

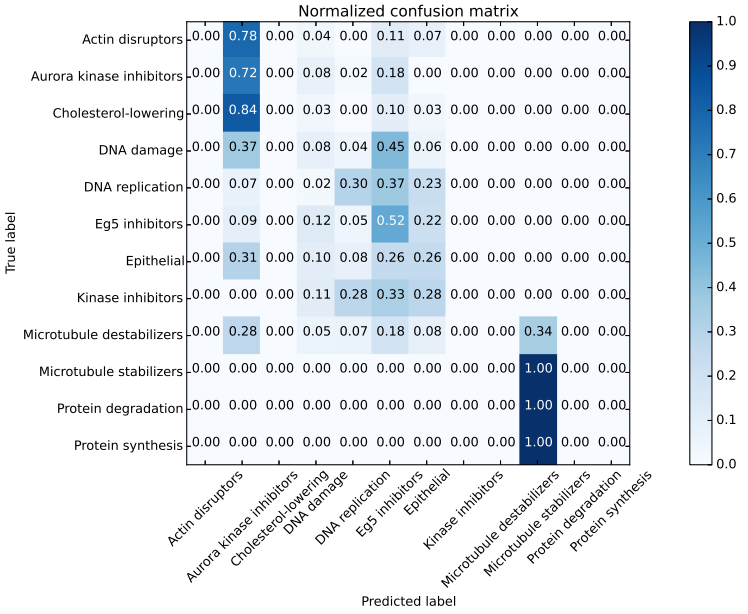

Figure 4.20: normalized confusion matrix, our VGG-11, imbalanced dataset.

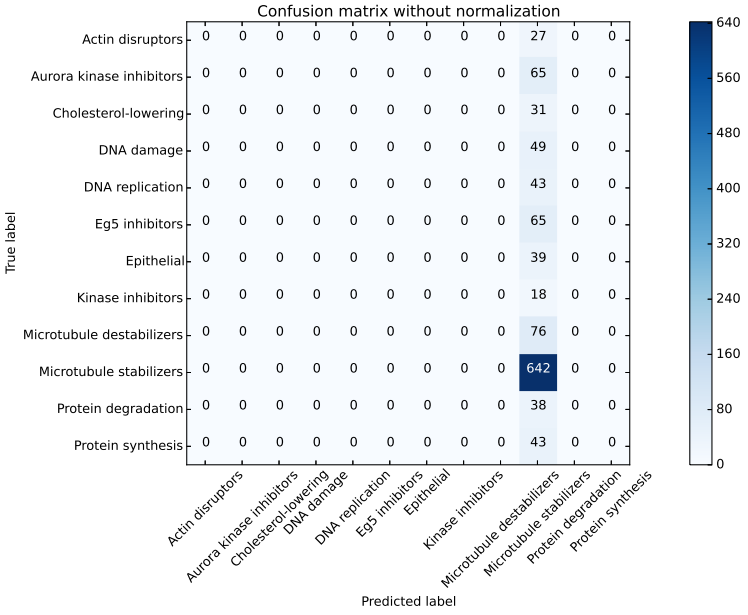

Figure 4.21: confusion matrix, our VGG-16, imbalanced dataset

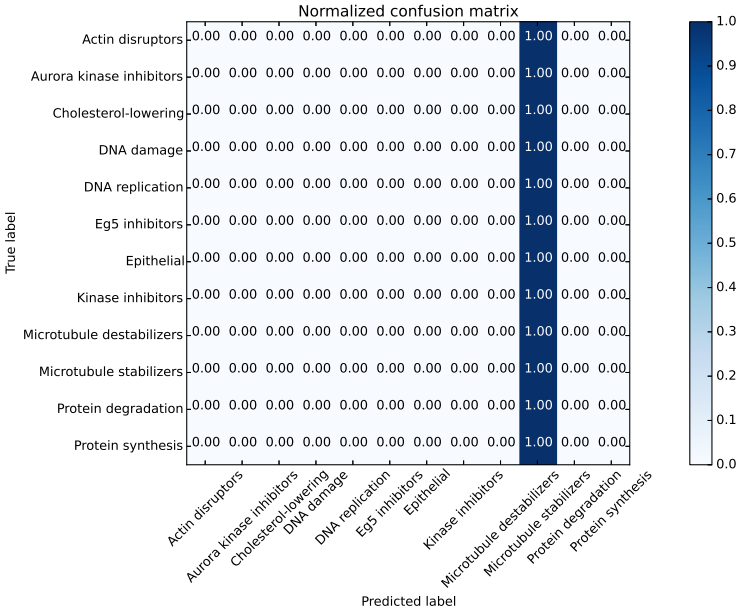

Figure 4.22: normalized confusion matrix, our VGG-16, imbalanced dataset.

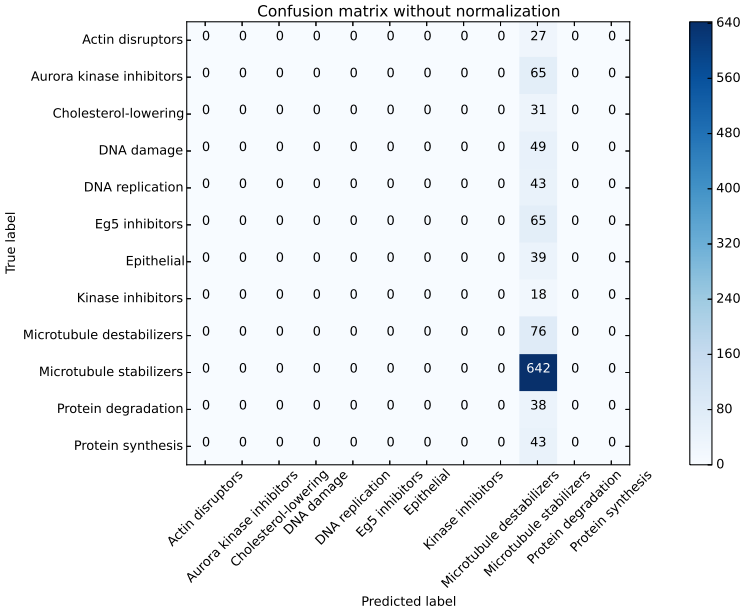

Figure 4.23: confusion matrix, our VGG-19, imbalanced dataset

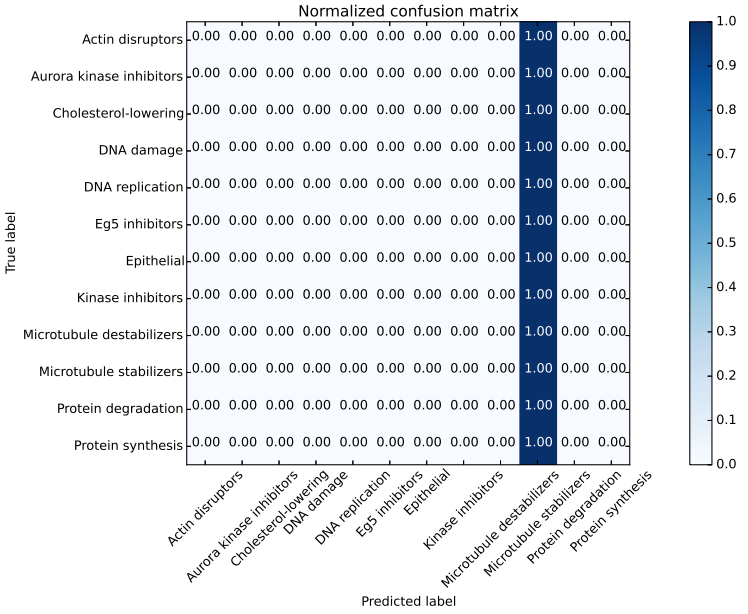

Figure 4.24: normalized confusion matrix, our VGG-19, imbalanced dataset.

4.4.2 Combination of our Deep Learning models and SMOTE algorithm

Table 4.5 shows the number of samples in the balanced dataset, train dataset, and test dataset by using SMOTE algorithm. Based on the results in Figures 4.25 to 4.38, Tables 4.6, and 4.7, many protein degradation and protein synthesis were misclassified as microtubule stabilizers MOA when SMOTE and our models including GoogleNet or VGG-11 or VGG-16 were combined. Moreover, many microtubule stabilizers and protein degradation were still misclassified as protein synthesis when SMOTE and AlexNet were combined. Many microtubule stabilizers and protein synthesis were misclassified as protein degradation when SMOTE and VGG-19 were combined (Figure 4.38).

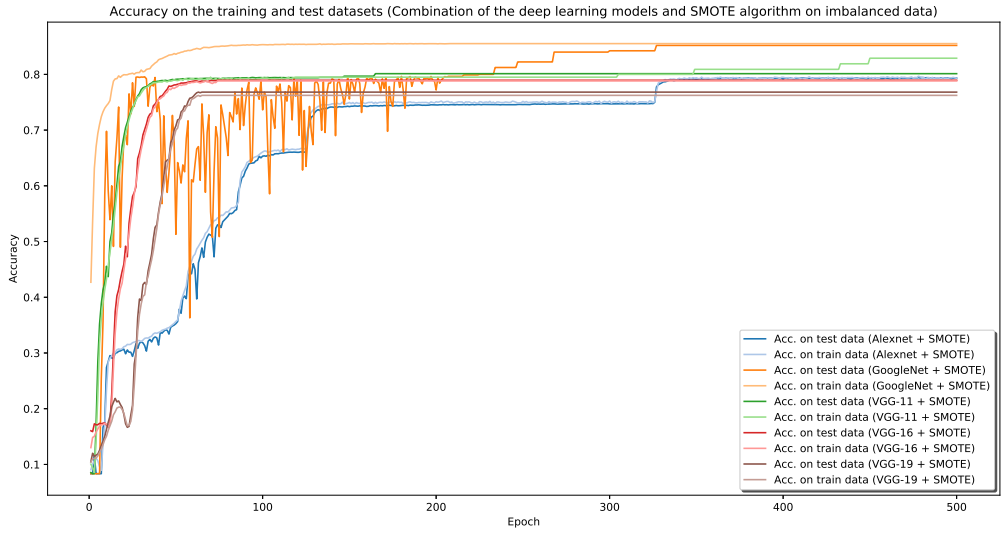

Figure 4.25: Accuracy on the training and test datasets (combination of the deep learning models and SMOTE algorithm on the imbalanced data)

Table 4.5: The number of samples in the balanced dataset by SMOTE is 51408. The number of samples in the train dataset is 43696. The number of samples in the test dataset is 7712.

MOA (balanced dataset-SMOTE)	Code of class	Number of samples	
		train dataset	test dataset
Actin disruptors	0	3641	643
Aurora kinase inhibitors	1	3641	643
Cholesterol-lowering	2	3642	642
DNA damage	3	3641	643
DNA replication	4	3642	642
Eg5 inhibitors	5	3641	643
Epithelial	6	3641	643
Kinase inhibitors	7	3642	642
Microtubule destabilizers	8	3641	643
Microtubule stabilizers	9	3641	643
Protein degradation	10	3642	642
Protein synthesis	11	3641	643

Table 4.6: The accuracy on the balanced test dataset by using SMOTE.

	Our models inspired by				
	GoogleNet	AlexNet	VGG-16 (VGG-11)	VGG-16	VGG-19
Accuracy	86%	79%	80%	78%	76%

Table 4.7: The top-2 and top-3 errors on the balanced test dataset by using SMOTE.

	Our models inspired by				
	GoogleNet	AlexNet	VGG-16 (VGG-11)	VGG-16	VGG-19
Top-2 error	0.11	0.12	0.12	0.12	0.13
Top-3 error	0.03	0.03	0.03	0.04	0.05

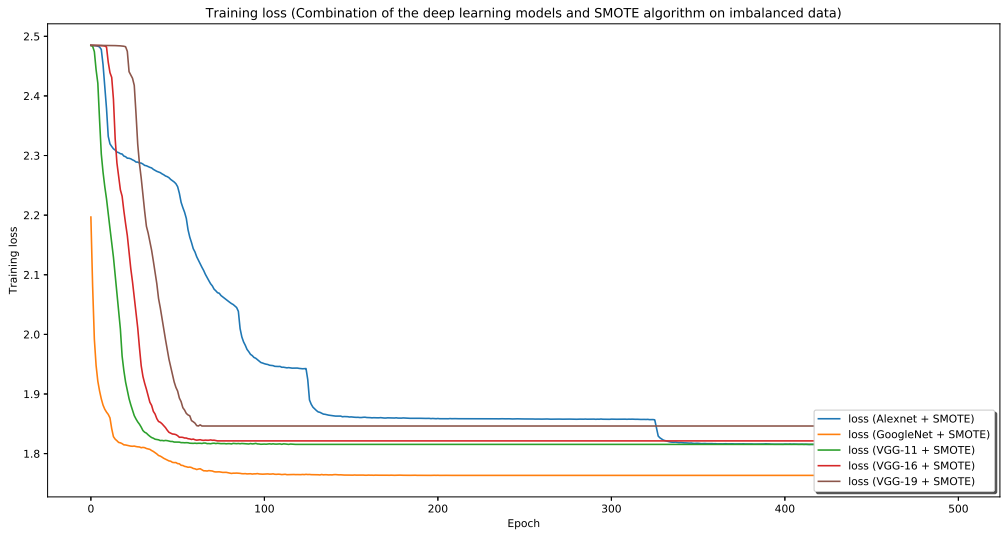

Figure 4.26: Training loss (combination of the deep learning models and SMOTE algorithm on the imbalanced data)

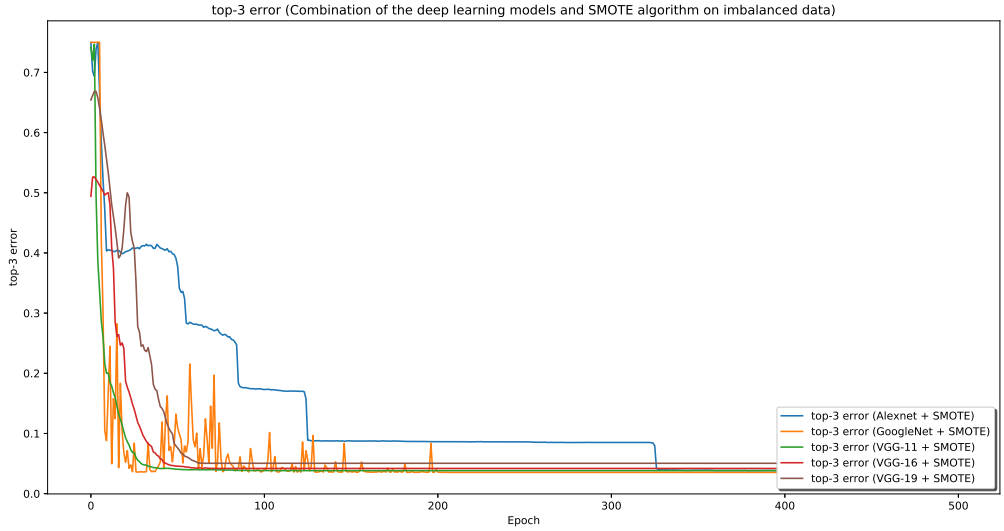

Figure 4.27: top-3 error on the balanced dataset by using SMOTE

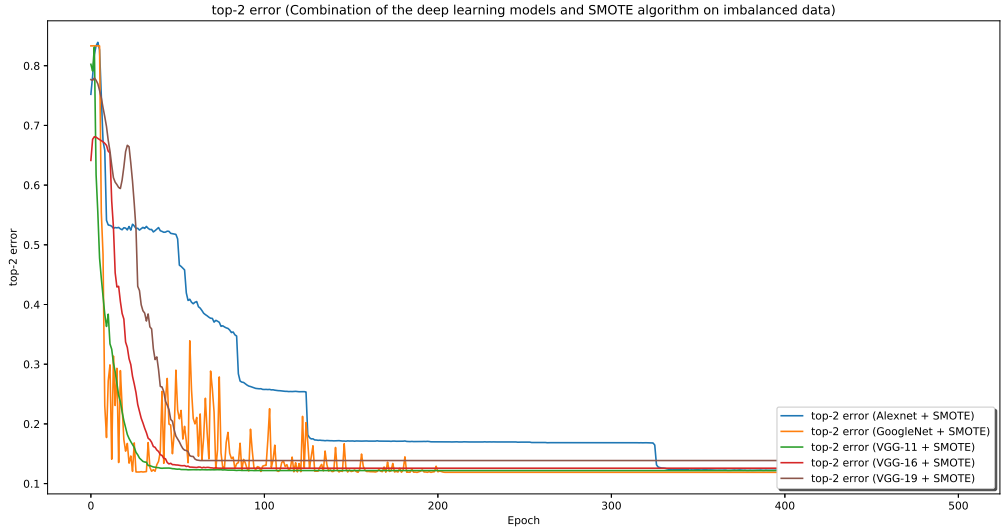

Figure 4.28: top-2 error by using SMOTE

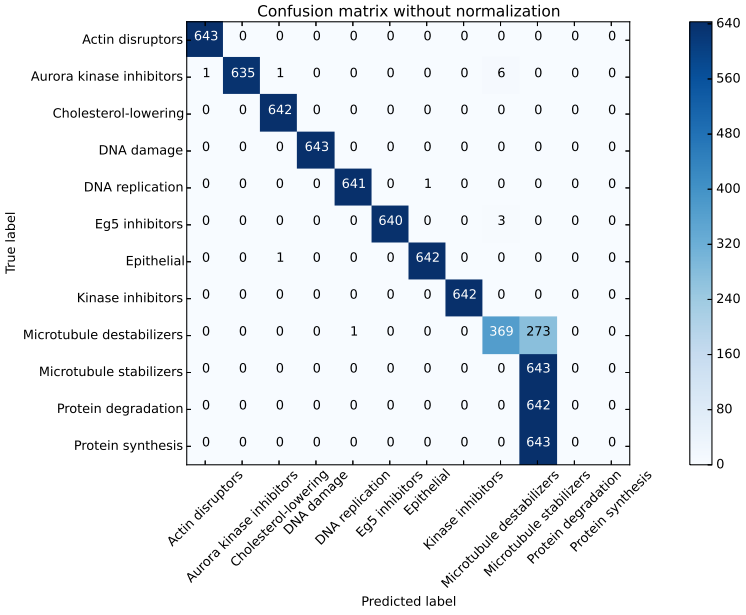

Figure 4.29: confusion matrix, our GoogleNet, SMOTE,balanced dataset.

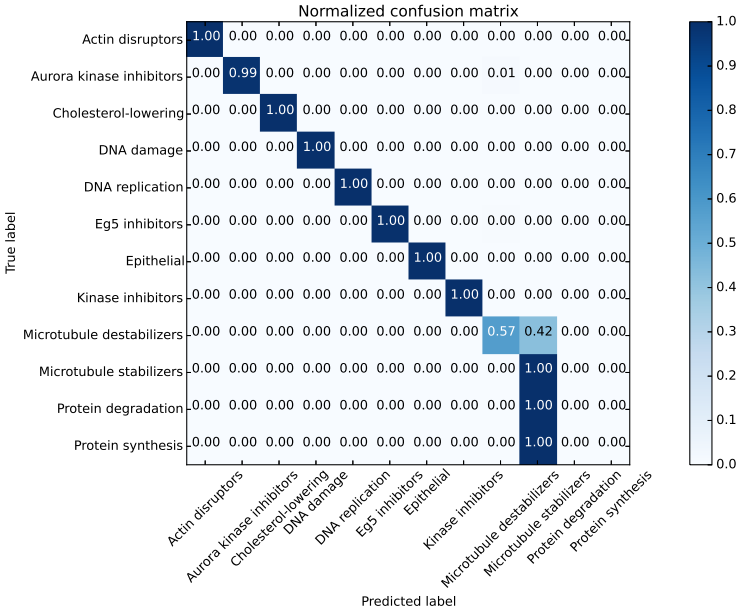

Figure 4.30: normalized confusion matrix, our GoogleNet,SMOTE,balanced dataset.

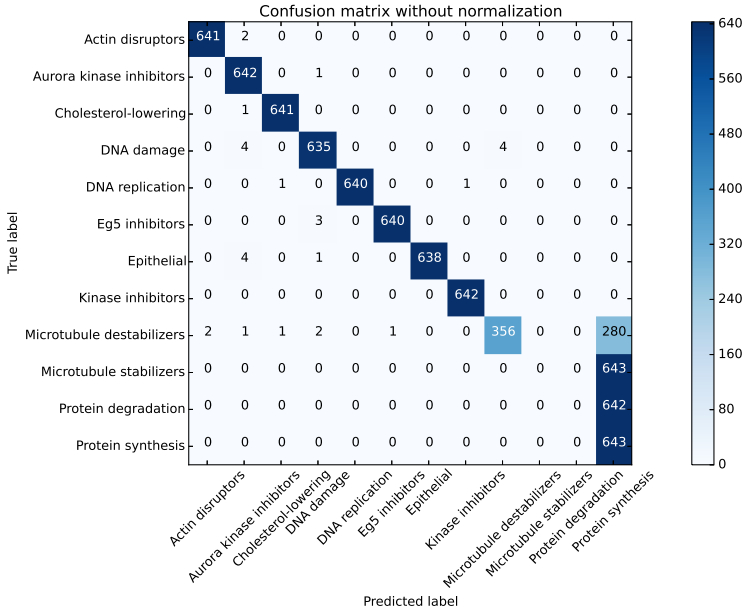

Figure 4.31: confusion matrix, our AlexNet,SMOTE,balanced dataset.

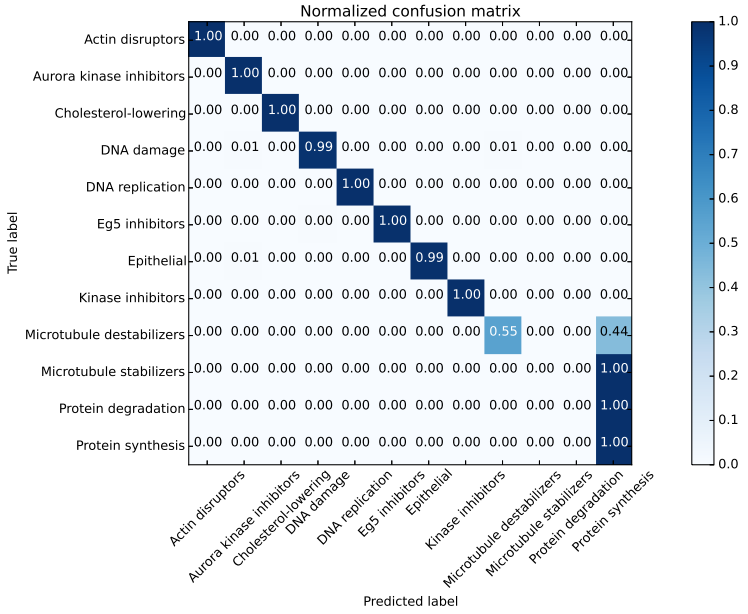

Figure 4.32: normalized confusion matrix, our AlexNet,SMOTE,balanced dataset.

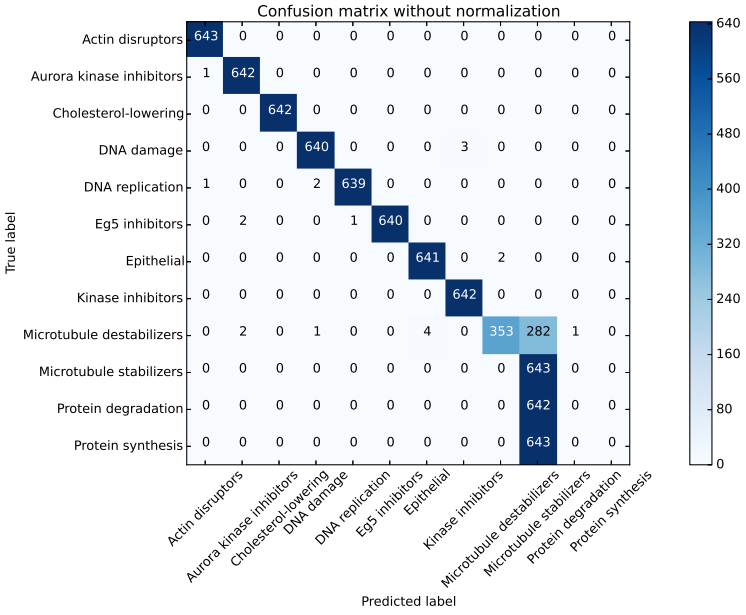

Figure 4.33: confusion matrix, our VGG-11,SMOTE,balanced dataset.

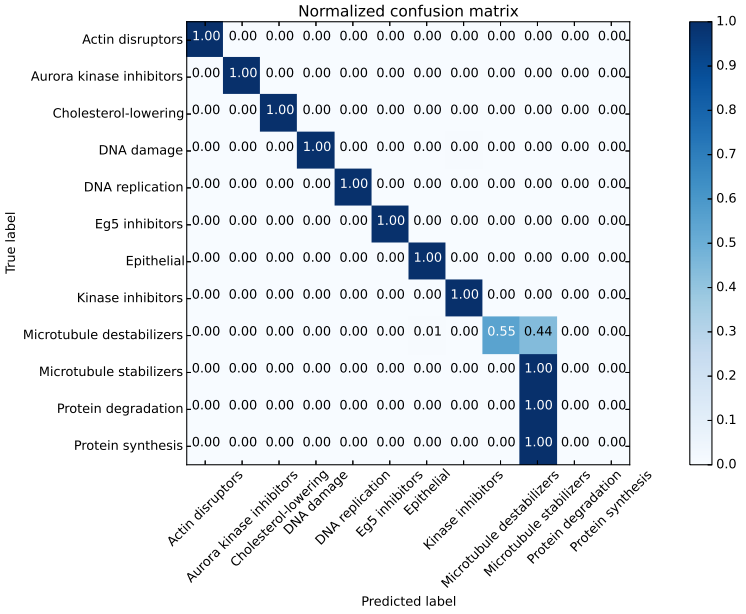

Figure 4.34: normalized confusion matrix, our VGG-11,SMOTE,balanced dataset.

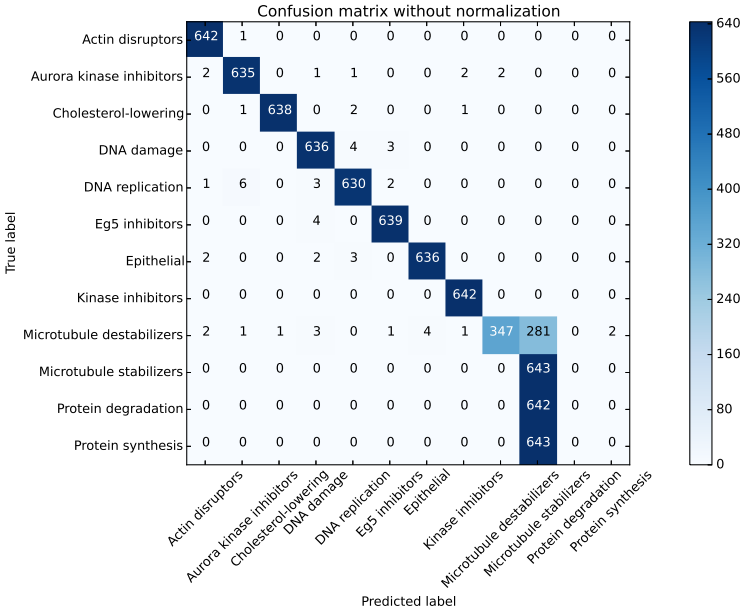

Figure 4.35: confusion matrix, our VGG-16,SMOTE,balanced dataset.

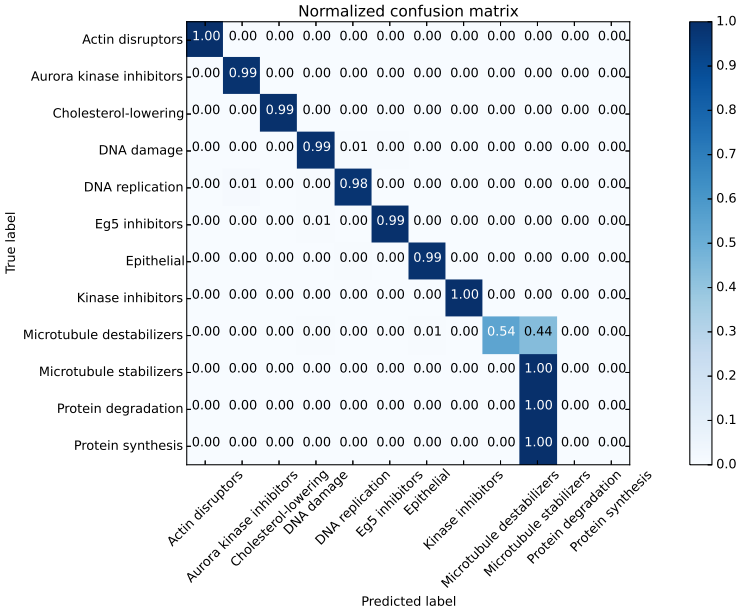

Figure 4.36: normalized confusion matrix, our VGG-16,SMOTE,balanced dataset.

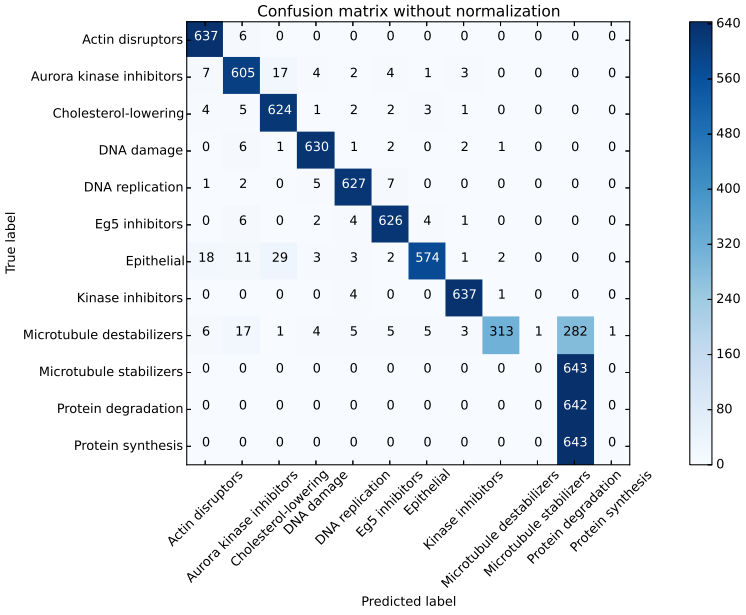

Figure 4.37: confusion matrix, our VGG-19,SMOTE,balanced dataset.

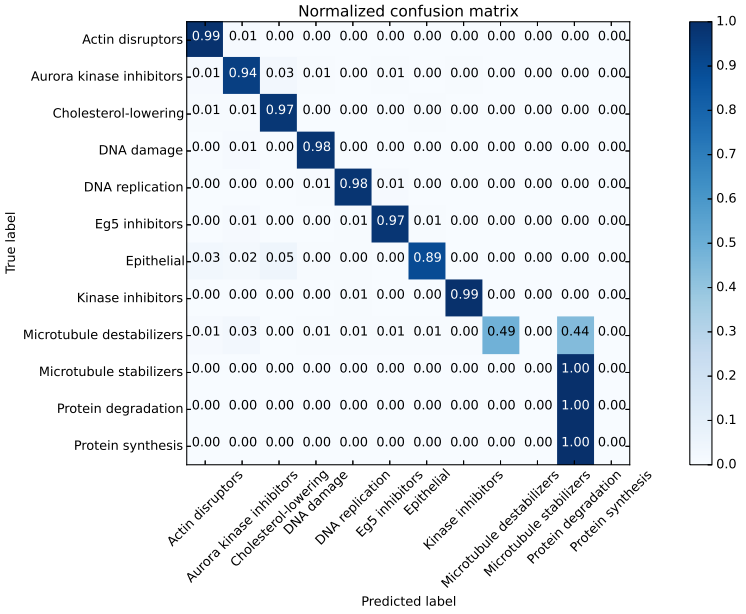

Figure 4.38: normalized confusion matrix, our VGG-19,SMOTE,balanced dataset.

4.4.3 Combination of our Deep Learning models and Undersampling algorithm

The results are shown in Figures 4.39 to 4.52, Tables 4.9, and 4.10. Table 4.8 shows the number of samples for each MOA class and the number of samples as train and test samples in the balanced dataset by using Undersampling method. Many microtubule stabilizers and protein synthesis were misclassified as protein degradation when Undersampling and our models including AlexNet, VGG-11, and VGG-16 were combined (Figures 4.45, 4.46, 4.47, 4.48, 4.49, 4.50). Protein degradation and protein synthesis were misclassified as microtubule stabilizers when Undersampling and GoogleNet were combined (Figures 4.43, 4.44).

Table 4.8: The number of samples in the balanced dataset by Undersampling method is 1440. The number of samples in the train dataset is 1224. The number of samples in the test dataset is 216.

MOA (balanced dataset-Undersampling)	Number of samples	
	train dataset	test dataset
For each MOA class	102	18

Table 4.9: The accuracy on balanced test dataset by using Undersampling method.

	Our models inspired by				
	GoogleNet	AlexNet	VGG-16 (VGG-11)	VGG-16	VGG-19
Accuracy	45%	31%	26%	18%	22%

Table 4.10: The top-2 and top-3 errors on the balanced test dataset by using Undersampling method.

	Our models inspired by				
	GoogleNet	AlexNet	VGG-16 (VGG-11)	VGG-16	VGG-19
Top-2 error	0.33	0.5	0.5	0.63	0.58
Top-3 error	0.17	0.30	0.30	0.42	0.41

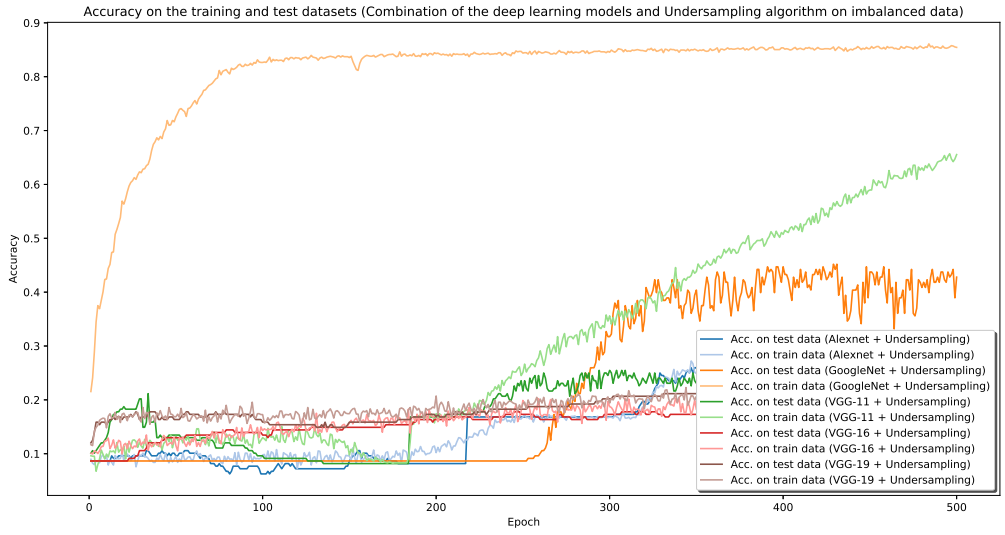

Figure 4.39: Accuracy on the training and test datasets (combination of the deep learning models and Undersampling algorithm on the imbalanced data)

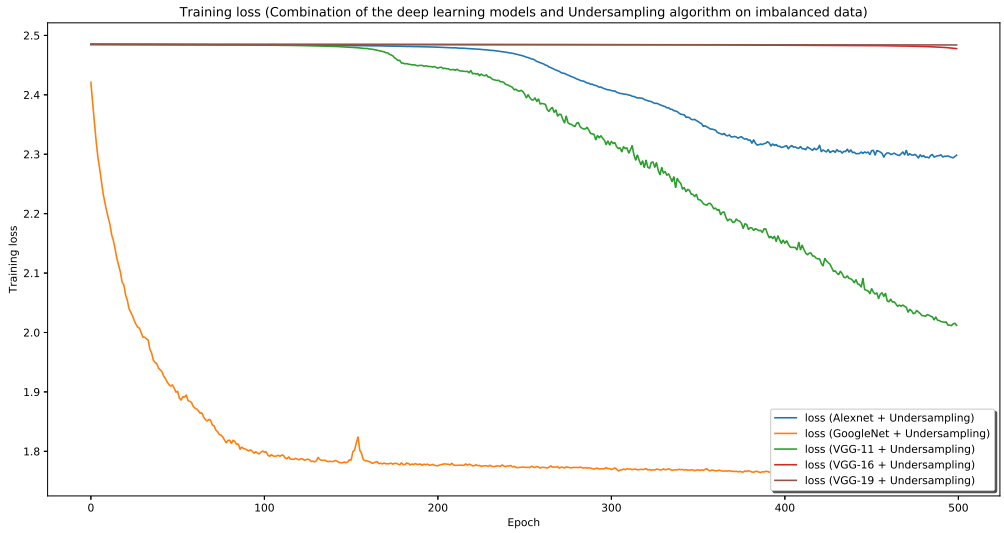

Figure 4.40: Training loss (combination of the deep learning models and Undersampling algorithm on the imbalanced data)

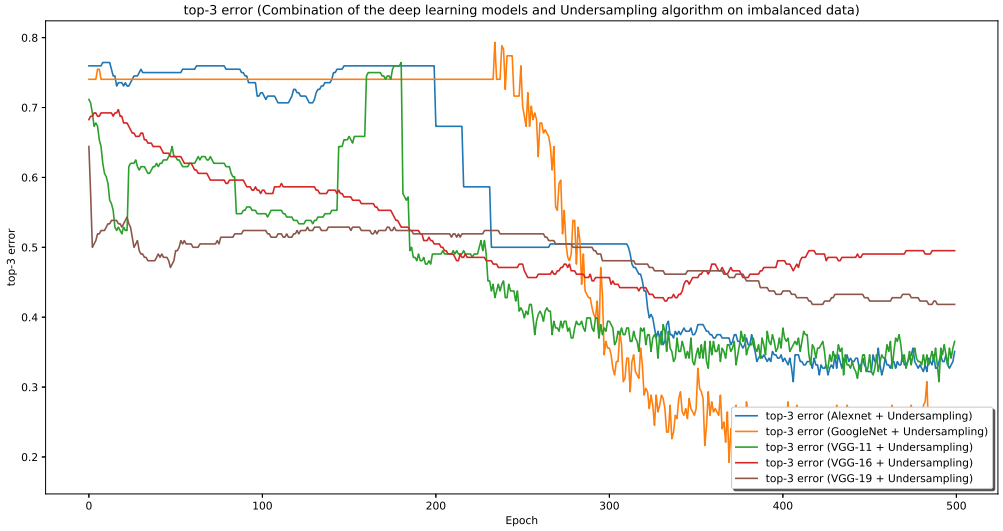

Figure 4.41: top-3 error by using Undersampling.

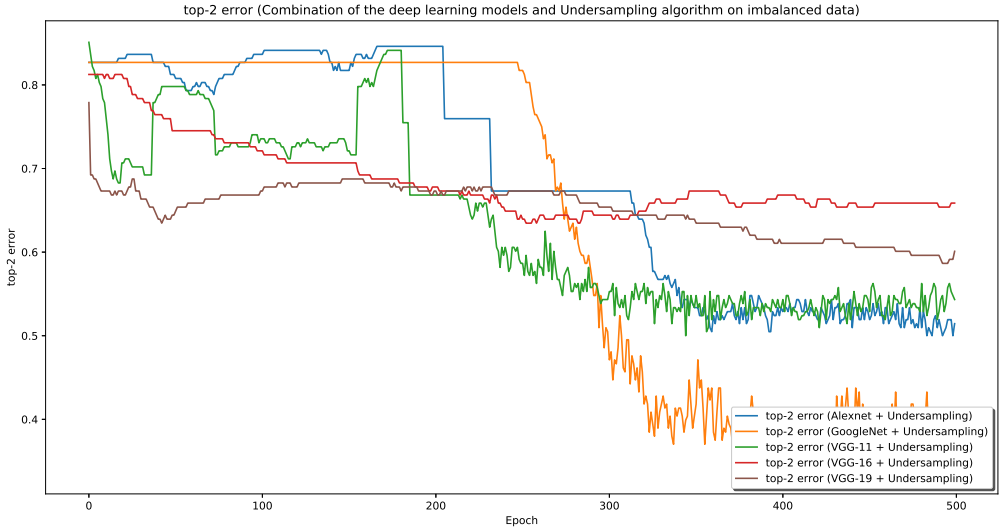

Figure 4.42: top-2 error by using Undersampling

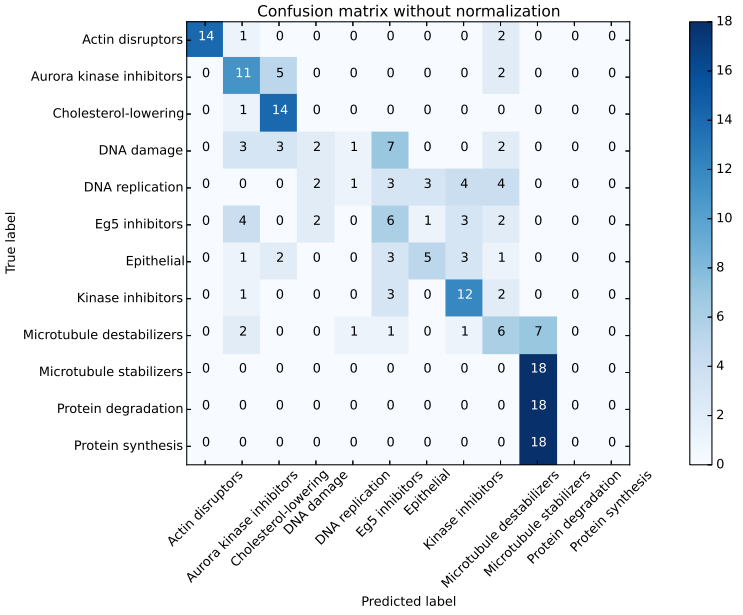

Figure 4.43: confusion matrix, our GoogleNet, Undersampling.

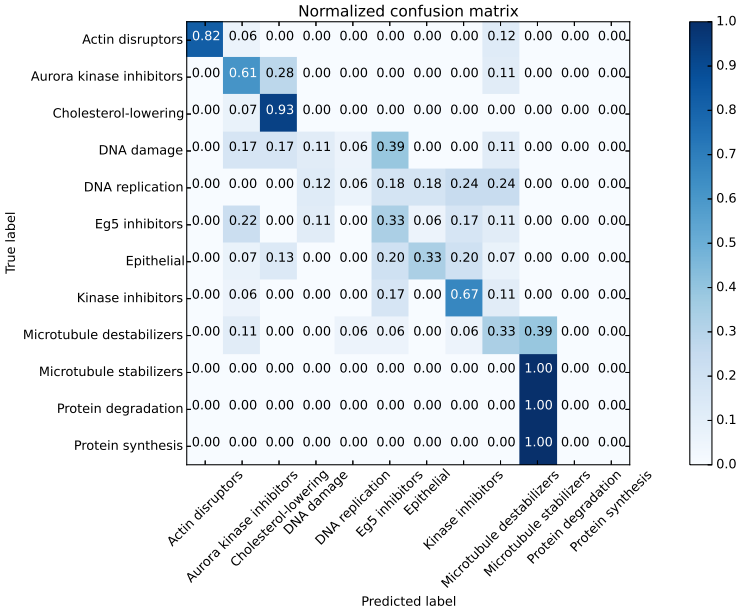

Figure 4.44: normalized confusion matrix, our GoogleNet, Undersampling.

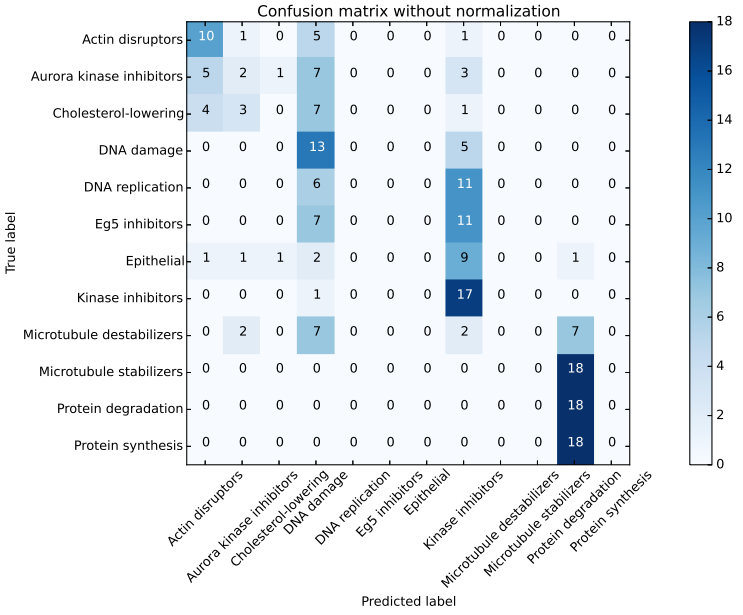

Figure 4.45: confusion matrix, our AlexNet, Undersampling.

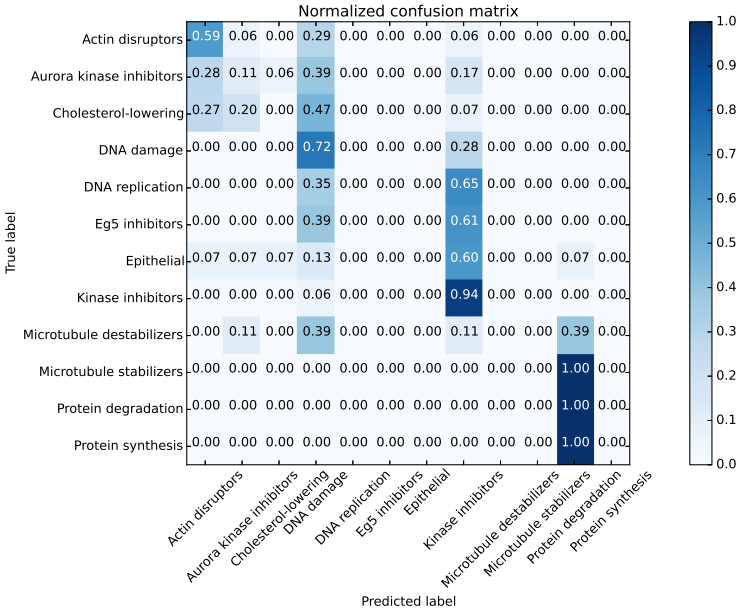

Figure 4.46: normalized confusion matrix, our AlexNet, Undersampling.

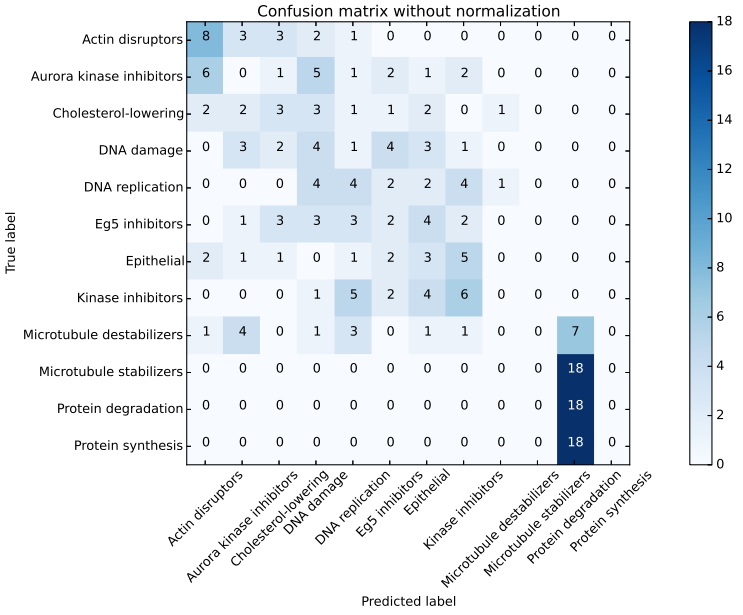

Figure 4.47: confusion matrix, our VGG-11, Undersampling.

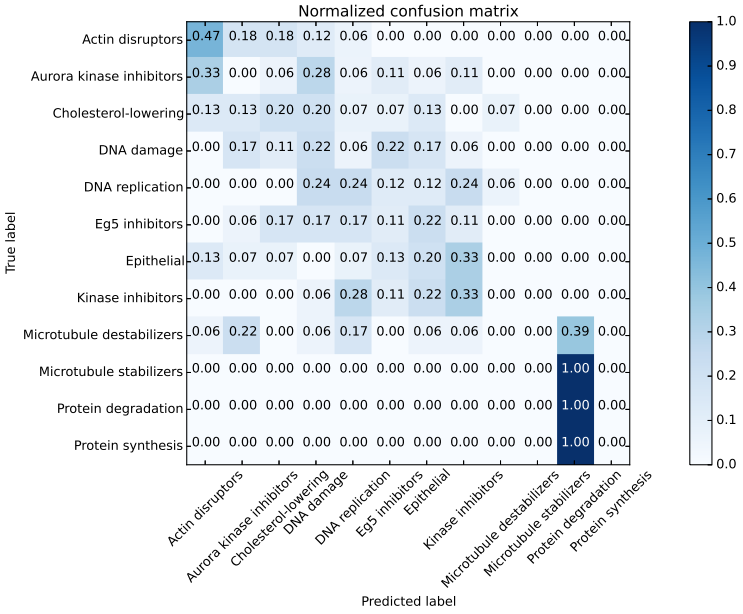

Figure 4.48: normalized confusion matrix, our VGG-11, Undersampling.

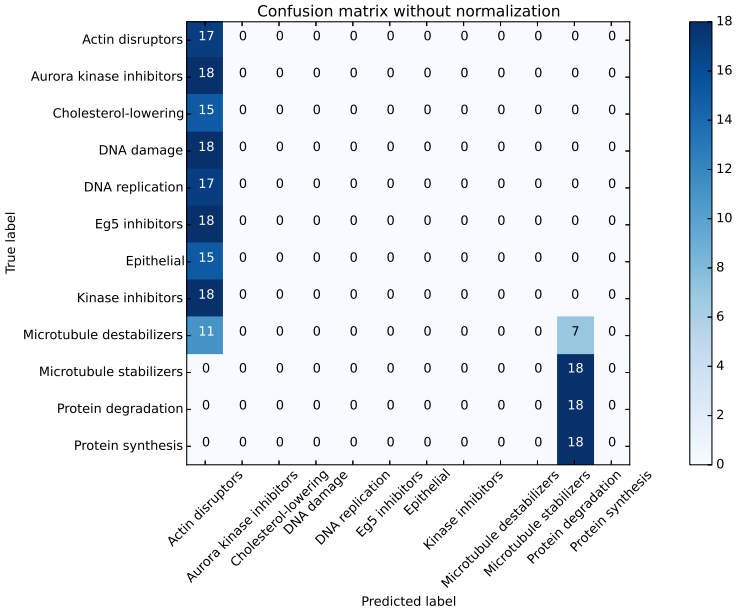

Figure 4.49: confusion matrix, our VGG-16, Undersampling.

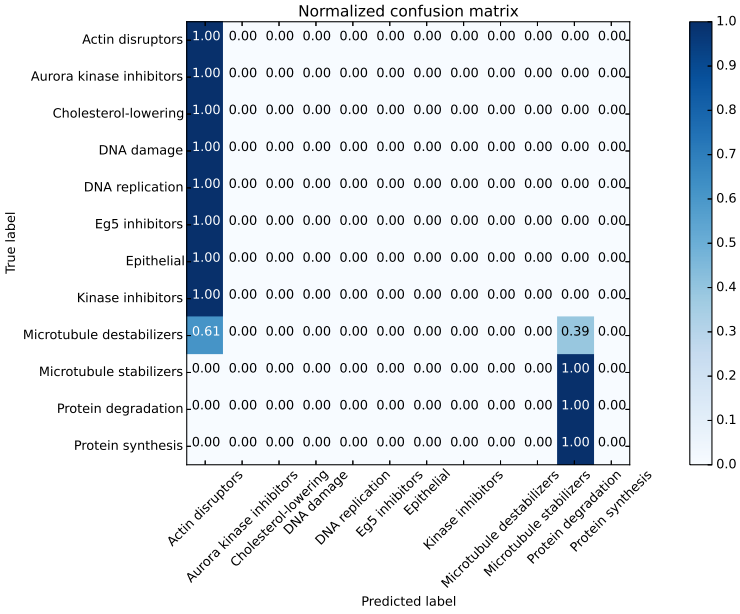

Figure 4.50: normalized confusion matrix, our VGG-16, Undersampling.

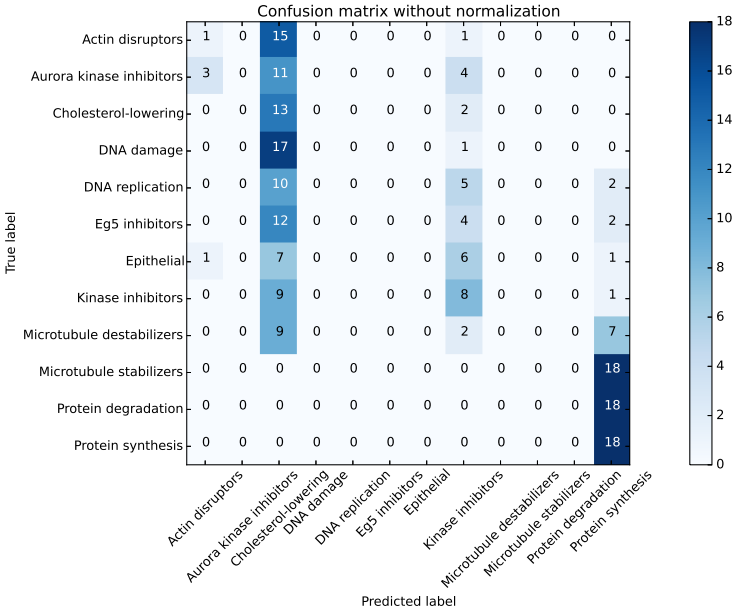

Figure 4.51: confusion matrix, our VGG-19, Undersampling.

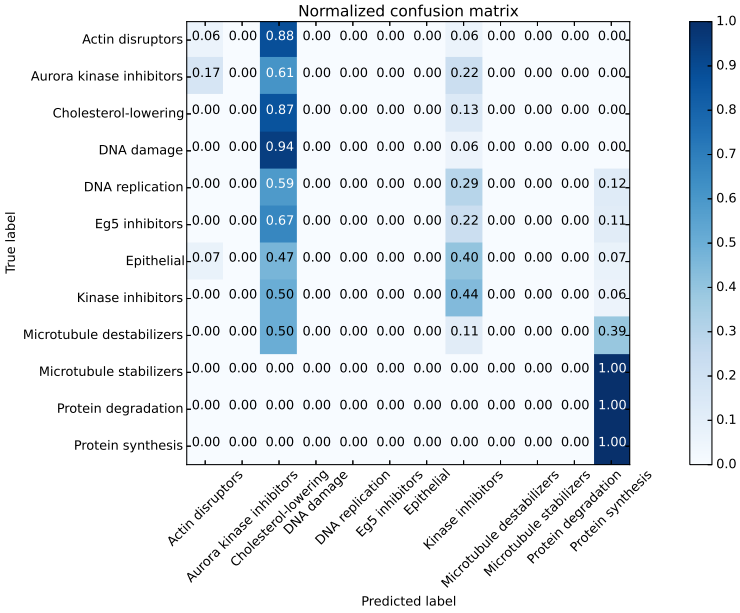

Figure 4.52: normalized confusion matrix, our VGG-19, Undersampling.

4.5 Comparison with the Previous Works

Although our models were performed on a population of cells for predicting mechanism of actions in drug discovery, our models also were applied on single cell identification because of comparing our methodologies performance with the results of Ljosa et al. [47], Kandaswamy et al. [36], Kraus et al. [39], Ando et al. [2], and Singh et al. [60]. The previous works did not test their methods on raw images(gray scale images) directly in contrast to our approaches. Ljosa et al. [47] used 453 features obtained from applying CellProfiler software. Kandaswamy et al. [36] used the 453 features of each cell, similar to the work of Ljosa [47] through CellProfiler software for a particular MOA class. Kraus et al. [39] also used CellProfiler software to get the segmented cell images. In addition, they proposed a method for localizing cells with Jacobian maps for detecting cells in the image. Ando et al. [2] demonstrated that the small total variation PCA dimensions are important for identifying cellular phenotypes, while the large total variation PCA dimensions are actually more representative of nuisance variation. Moreover, Ando et al. [2], Ljosa et. al. [47], and Kandaswamy et al. [36] did not take advantage of the automatic feature extraction property as one of the most important features in deep learning while this most important characteristic was used perfectly in our models.

In contrast to previous methods, our models outperformed previously published results for MCF-7 breast cancer cells without any pre or post processing modules except normalizing the images by subtracting the mean and dividing by the standard deviation of each channel in our training set. Our results indicated that subtracting the mean and dividing by the standard deviation process does not provide significant

improvement in accuracy of our models.

In contrast to previous works, our models were tested on raw grayscale images in the BBBC021 dataset. Previous works only used a small subset of the BBBC021 dataset as training and test datasets. Not only our models were tested on all true labeled samples in BBBC021 dataset (as explained in previous sections), but also, in order to compare with previous works, our models were tested on the same subset of the BBBC021 dataset that Ljosa et al.[47], Kandaswamy et al.[36], Kraus et al.[39], and Ando et al.[2] used. Thus we considered 15% of images from these 103 treatments to train and validate our models. The similar proportion of the data reported in previous works was applied to train our best model. We evaluated all the image channels and reported the predicted accuracy across the treatments. According to our experiments, we believe that using this small volume dataset without dropout can lead to overfitting for all deep learning models. Thus, as noted in our methodology section(chapter 3), dropout was the main element that we used in input and final layers for avoiding overfitting. In comparison to the previous works, our GoogleNet and VGG-11 models combined with SMOTE method achieved better accuracy(99% accuracy and 98% accuracy respectively in Table 4.11). The center of mass coordinates of segmented cells were extracted and these coordinates were applied to crop single cells from the full resolution images using CellProfiler. The crop size was supposed to be 64*64 pixels.

Our model inspired by GoogleNet exceeds the newest accuracy with 99% NSC(Not-Same-Compound) accuracy. The accuracy for 10 MOAs of the 12 MOAs is 100% with the error in classifying protein degradation and protein synthesis MOAs (normalized confusion matrices in Figures 4.30, 4.32, 4.34, 4.36).

Table 4.11: Comparison with the previous works (Predicting MOAs based on a single cell view.).

Method	Accuracy
Ljosa et al. [47] (2013), traditional method	94%
Singh et al. [60] (2014), traditional method	90%
Kandaswamy et al.[36], (2016)	88%
Kraus et al.[39], (2016)	97%
Ando et al.[2], (2017)	96%
Our model (SMOTE + GoogleNet)	99%
Our model (SMOTE + VGG-11)	98%

Chapter 5

Conclusions and Future Work

This work proposes a fully automated system using deep learning without human interaction. We have shown that deep neural networks are capable of recognizing MOAs from microscopy images if they were configured properly. The proposed models achieve higher accuracies than previous classical methods, requires less time and expertise to recognize MOAs because image segmentation and feature extraction modules are eliminated. This work establishes the basis of High Content Screening systems for future explorations.

This system does not need to use CellProfiler software for detecting and preprocessing cell images. We actually deployed multi-object detection and recognition automatically without using object detection, preprocessing and feature extraction steps for predicting MOAs in drug discovery. Not only our deep learning models worked on a population of cells very well for predicting MOAs but also they achieved better accuracy in comparison to the previous works.

Bibliography

[1] E. Alpaydin. Introduction to machine learning, 2nd edn. adaptive computation and machine learning, 2010.

[2] D. M. Ando, C. McLean, and M. Berndl. Improving phenotypic measurements in high-content imaging screens. *bioRxiv*, page 161422, 2017.

[3] C. Bakal, J. Aach, G. Church, and N. Perrimon. Quantitative morphological signatures define local signaling networks regulating cell morphology. *science*, 316(5832):1753–1756, 2007.

[4] N. Battich, T. Stoeger, and L. Pelkmans. Image-based transcriptomics in thousands of single human cells at single-molecule resolution. *Nature methods*, 10(11):1127–1133, 2013.

[5] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[6] C. M. Bishop. Pattern recognition. *Machine Learning*, 128, 2006.

[7] M. V. Boland and R. F. Murphy. A neural network classifier capable of recognizing the patterns of all major subcellular structures in fluorescence microscope images of hela cells. *Bioinformatics*, 17(12):1213–1223, 2001.

[8] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] M. Breker, M. Gymrek, and M. Schuldiner. A novel single-cell screening platform reveals proteome plasticity during yeast stress responses. *The Journal of cell biology*, 200(6):839–850, 2013.

[10] 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*, 9(6):1913–1926, 2010.

[11] 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, et al. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome biology*, 7(10):R100, 2006.

[12] A. E. Carpenter and D. M. Sabatini. Systematic genome-wide screens of gene function. *Nature Reviews Genetics*, 5(1):11–22, 2004.

[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[14] S.-C. Chen, T. Zhao, G. J. Gordon, and R. F. Murphy. Automated image analysis of protein localization in budding yeast. *Bioinformatics*, 23(13):i66–i71, 2007.

[15] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan. Aircraft detection by deep belief nets. In *2nd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 54–58. IEEE, 2013.

[16] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan. Vehicle detection in satellite images by hybrid deep convolutional neural networks. *Geoscience and Remote Sensing Letters, IEEE*, 11(10):1797–1801, 2014.

[17] Y. T. Chong, J. L. Koh, H. Friesen, S. K. Duffy, M. J. Cox, A. Moses, J. Moffat, C. Boone, and B. J. Andrews. Yeast proteome dynamics from single cell imaging and automated analysis. *Cell*, 161(6):1413–1424, 2015.

[18] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[19] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[20] K. W. Eliceiri, M. R. Berthold, I. G. Goldberg, L. Ibáñez, B. S. Manjunath, M. E. Martone, R. F. Murphy, H. Peng, A. L. Plant, B. Roysam, et al. Biological imaging software tools. *Nature methods*, 9(7):697–710, 2012.

[21] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013.

[22] Y. Freund, R. E. Schapire, et al. Experiments with a new boosting algorithm. In *Icml*, volume 96, pages 148–156. Bari, Italy, 1996.

[23] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007.

[24] F. Fuchs, G. Pau, D. Kranz, O. Sklyar, C. Budjan, S. Steinbrink, T. Horn, A. Pedal, W. Huber, and M. Boutros. Clustering phenotype populations by genome-wide rnai and multiparametric imaging. *Molecular systems biology*, 6(1):370, 2010.

[25] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, et al. Bioconductor: open software development for computational biology and bioinformatics. *Genome biology*, 5(10):1, 2004.

[26] Z. Ghahramani. Unsupervised learning. In *Advanced lectures on machine learning*, pages 72–112. Springer, 2004.

[27] S. A. Haney. *High Content Screening: Science, Techniques and Applications*. John Wiley & Sons, 2008.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[29] M. Held, M. H. Schmitz, B. Fischer, T. Walter, B. Neumann, M. H. Olma, M. Peter, J. Ellenberg, and D. W. Gerlich. Cellcognition: time-resolved phenotype annotation in high-throughput live cell imaging. *Nature methods*, 7(9):747–754, 2010.

[30] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[31] P. Horvath, T. Wild, U. Kutay, and G. Csucs. Machine learning improves the precision and robustness of high-content screens using nonlinear multiparametric methods to analyze screening results. *Journal of biomolecular screening*, 16(9):1059–1067, 2011.

[32] D. Houle, D. R. Govindaraju, and S. Omholt. Phenomics: the next challenge. *Nature Reviews Genetics*, 11(12):855–866, 2010.

[33] J. E. Jackson. *A user's guide to principal components*, volume 587. John Wiley & Sons, 2005.

[34] I. T. Jolliffe. Principal component analysis and factor analysis. *Principal component analysis*, pages 150–166, 2002.

[35] T. R. Jones, I. H. Kang, D. B. Wheeler, R. A. Lindquist, A. Papallo, D. M. Sabatini, P. Golland, and A. E. Carpenter. Cellprofiler analyst: data exploration

and analysis software for complex image-based screens. *BMC bioinformatics*, 9(1):1, 2008.

[36] C. Kandaswamy, L. M. Silva, L. A. Alexandre, and J. M. Santos. High-content analysis of breast cancer using single-cell deep transfer learning. *Journal of biomolecular screening*, page 1087057115623451, 2016.

[37] V. Kaynig, T. Fuchs, and J. M. Buhmann. Neuron geometry extraction by perceptual grouping in sstem images. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2902–2909. IEEE, 2010.

[38] S. J. Kiddle, O. P. Windram, S. McHattie, A. Mead, J. Beynon, V. Buchanan-Wollaston, K. J. Denby, and S. Mukherjee. Temporal clustering by affinity propagation reveals transcriptional modules in arabidopsis thaliana. *Bioinformatics*, 26(3):355–362, 2010.

[39] O. Z. Kraus, J. L. Ba, and B. J. Frey. Classifying and segmenting microscopy images with deep multiple instance learning. *Bioinformatics*, 32(12):i52–i59, 2016.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[41] K. L. Kurita, E. Glassey, and R. G. Linington. Integration of high-content screening and untargeted metabolomics for comprehensive functional annotation of natural product libraries. *Proceedings of the National Academy of Sciences*, 112(39):11999–12004, 2015.

[42] P. Lang, K. Yeow, A. Nichols, and A. Scheer. Cellular imaging in drug discovery. *Nature Reviews Drug Discovery*, 5(4):343–356, 2006.

[43] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[44] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[45] M. Leone, M. Weigt, et al. Clustering by soft-constraint affinity propagation: applications to gene-expression data. *Bioinformatics*, 23(20):2708–2715, 2007.

[46] P. Liberali, B. Snijder, and L. Pelkmans. Single-cell and multivariate approaches in genetic perturbation screens. *Nature Reviews Genetics*, 16(1):18–32, 2015.

[47] V. Ljosa, P. D. Caie, R. ter Horst, K. L. Sokolnicki, E. L. Jenkins, S. Daya, M. E. Roberts, T. R. Jones, S. Singh, A. Genovesio, et al. Comparison of methods for image-based profiling of cellular morphological responses to small-molecule treatment. *Journal of biomolecular screening*, page 1087057113503553, 2013.

[48] V. Ljosa, K. L. Sokolnicki, and A. E. Carpenter. Annotated high-throughput microscopy image sets for validation. *Nat Methods*, 9(7):637, 2012.

[49] L.-H. Loo, L. F. Wu, and S. J. Altschuler. Image-based multivariate profiling of drug responses from single cells. *Nature methods*, 4(5):445–453, 2007.

[50] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[51] J. G. Moffat, J. Rudolph, and D. Bailey. Phenotypic screening in cancer drug discoverypast, present and future. *Nature Reviews Drug Discovery*, 13(8):588–602, 2014.

[52] K. Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[53] Z. E. Perlman, M. D. Slack, Y. Feng, T. J. Mitchison, L. F. Wu, and S. J. Altschuler. Multidimensional drug profiling by automated microscopy. *Science*, 306(5699):1194–1198, 2004.

[54] M. A. Ranzato, J. Susskind, V. Mnih, and G. Hinton. On deep generative models with applications to recognition. In *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2857–2864. IEEE, 2011.

[55] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[56] G. Saporta and N. Niang. Principal component analysis: application to statistical process control. *Data analysis*, pages 1–23, 2009.

[57] O. Shalem, N. E. Sanjana, and F. Zhang. High-throughput functional genomics using crispr-cas9. *Nature Reviews Genetics*, 16(5):299–311, 2015.

[58] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[59] D. K. Singh, C.-J. Ku, C. Wichaidit, R. J. Steininger, L. F. Wu, and S. J. Altschuler. Patterns of basal signaling heterogeneity can distinguish cellular populations with different drug sensitivities. *Molecular systems biology*, 6(1):369, 2010.

[60] S. Singh, M.-A. BRAY, T. Jones, and A. Carpenter. Pipeline for illumination correction of images for high-throughput microscopy. *Journal of microscopy*, 256(3):231–236, 2014.

[61] S. Singh, A. E. Carpenter, and A. Genovesio. Increasing the content of high-content screening an overview. *Journal of biomolecular screening*, 19(5):640–650, 2014.

[62] C. Sommer and D. W. Gerlich. Machine learning in cell biology–teaching computers to recognize phenotypes. *J Cell Sci*, 126(24):5529–5539, 2013.

[63] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.

[64] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[65] D. L. Taylor and J. R. Haskins. *High content screening: A powerful approach to systems cell biology and drug discovery*, volume 356. Springer Science & Business Media, 2007.

[66] J. M. Tkach, A. Yimit, A. Y. Lee, M. Riffle, M. Costanzo, D. Jaschob, J. A. Hendry, J. Ou, J. Moffat, C. Boone, et al. Dissecting dna damage response pathways by analysing protein localization and abundance changes during dna replication stress. *Nature cell biology*, 14(9):966–976, 2012.

[67] A. H. Y. Tong, M. Evangelista, A. B. Parsons, H. Xu, G. D. Bader, N. Page, M. Robinson, S. Raghibizadeh, C. W. Hogue, H. Bussey, et al. Systematic genetic analysis with ordered arrays of yeast deletion mutants. *Science*, 294(5550):2364–2368, 2001.

[68] A. H. Y. Tong, G. Lesage, G. D. Bader, H. Ding, H. Xu, X. Xin, J. Young, G. F. Berriz, R. L. Brost, M. Chang, et al. Global mapping of the yeast genetic interaction network. *science*, 303(5659):808–813, 2004.

[69] F. J. Vizeacoumar, N. Van Dyk, F. S. Vizeacoumar, V. Cheung, J. Li, Y. Sydorskyy, N. Case, Z. Li, A. Datti, C. Nislow, et al. Integrating high-throughput genetic interaction mapping and high-content screening to explore yeast spindle morphogenesis. *The Journal of cell biology*, 188(1):69–81, 2010.

[70] Z. Yin, A. Sadok, H. Sailem, A. McCarthy, X. Xia, F. Li, M. A. Garcia, L. Evans, A. R. Barr, N. Perrimon, et al. A screen for morphological complexity identifies regulators of switch-like transitions between discrete cell shapes. *Nature cell biology*, 15(7):860–871, 2013.

[71] D. W. Young, A. Bender, J. Hoyt, E. McWhinnie, G.-W. Chirn, C. Y. Tao, J. A. Tallarico, M. Labow, J. L. Jenkins, T. J. Mitchison, et al. Integrating high-

content screening and ligand-target prediction to identify mechanism of action. *Nature chemical biology*, 4(1):59–68, 2008.

[72] L. Zamparo and Z. Zhang. Deep autoencoders for dimensionality reduction of high-content screening data. *arXiv preprint arXiv:1501.01348*, 2015.

[73] Q. Zhong, A. G. Busetto, J. P. Fededa, J. M. Buhmann, and D. W. Gerlich. Unsupervised modeling of cell morphology dynamics for time-lapse microscopy. *nature methods*, 9(7):711–713, 2012.