T.C.

ALTINBAS UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

USING MACHINE LEARNING ALGORITHMS FOR ORTHOPEDIC CLASSIFICATION

Mohamedsadeq Abdulkhudhr ALSAEDI

Master of Science

Supervisor

Prof. Dr. Osman Nuri UÇAN

Istanbul, 2021

USING MACHINE LEARNING ALGORITHMS FOR ORTHOPEDIC CLASSIFICATION

by

Mohamedsadeq Abdulkhudhr ALSAEDI

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Master of Science

ALTINBAŞ UNIVERSITY

2021

The thesis titled “USING MACHINE LEARNING ALGORITHMS FOR ORTHOPEDIC CLASSIFICATION” prepared and presented by “Mohamedsadeq Abdulkhudhr ALSAEDI” was accepted as a Master of Science Thesis in Electrical and Computer Engineering.

Prof. Dr. Osman Nuri UÇAN

Supervisor

Thesis Defense Jury Members:

Prof. Dr. Osman Nuri UÇAN	School of Engineering and Architecture, Altinbas University	_____
Asst. Prof. Dr. Sefer KURNAZ	School of Engineering and Architecture, Altinbas University	_____
Prof. Dr. Mesut RAZBONYALI	Faculty of Engineering and Architecture, Maltepe University	_____

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Approval Date of Institute of Graduate studies:

____/____/____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Mohamedsadeq Abdulkhudhr ALSAEDI

DEDICATION

I would like to thank God Almighty for the strength of mind, health, strength, guidance, knowledge and skills to complete this study.

This thesis is sincerely dedicated to my parents. There are no words to describe what it means to me There is nothing I can repay for what you did to me. I will continue to do my best to fulfill your expectations.

Finally, I dedicated this to my wife, my brother, my sisters, my relatives, my friends and to the martyrs of Iraq who fell for the freedom of this country and they were the ones who encouraged me during this study.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Prof. Dr. Osman Nuri UÇAN for all support during my study. It's great pleasure to express my deepest gratitude to my friends who have shared with me best moments during my study for the Master degree.

ABSTRACT

USING MACHINE LEARNING ALGORITHMS FOR ORTHOPEDIC CLASSIFICATION

ALSAEDI, Mohamedsadeq Abdulkhudhr,

M.Sc., Electrical and Computer Engineering ,Altınbaş University,

Supervisor: Prof. Dr. Osman Nuri UÇAN

Date: 10/2021

Pages: 45

Prediction and classification are the two class of problems to which machine learning algorithms are applied. Prediction algorithm takes one or more feature values and predict outcome that is continuous. Classification algorithm takes one or more feature values to predict categorical outcome. The classification outcome of a target variable may be binary or multinomial. The Logistic Regression algorithm can have used for classification to model a categorical outcome and many of its application are found in medical field. Different machine learning algorithms can be applied on same dataset, however, which ML algorithm will comparatively prove better depends upon the nature of the data and preprocessing applied on the data. Here, three algorithms logistic regression, decision tree and K-NN are applied on the dataset and compared using confusion matrix.

Keywords: Machine Learning Techniques, Decision Tree, Logistic Regression, K-NN.

TABLE OF CONTENTS

Pages

LIST OF FIGURES

Pages

LIST OF TABLES

Pages

LIST OF ABBREVIATIONS

ML	:	Machine Learning
NLP	:	Natural Language Processing
AI	:	Artificial Intelligence
NB	:	Naive Bays
SVM	:	Support Victor Machine
LSI	:	Latin Semantic Index
GOM	:	Goals of Matrices

1. INTRODUCTION

Prediction and classification are the two class of problems to which machine learning algorithms are applied. Prediction algorithm takes one or more feature values and predict outcome that is continuous. Classification algorithm takes one or more feature values to predict categorical outcome. The classification outcome of a target variable may be binary or multinomial. The Logistic Regression algorithm can use for classification to model a categorical outcome [1] and many of its application are found in medical field [2]. Logistic regression is used extensively for binary classification but it also be used for multinomial classification [3]. Another machine learning algorithm which is widely used for classification is decision trees [4] [5], they are effective for binary classification but are also applicable to multiclass problem. Logistic regression is equation based where the feature variables (independent variables) in the equation are used to predict the dependent variables where as in decision trees it is rule based. The third machine learning model considered here to be applied on the dataset chosen is KNN [6]. KNN – K-Nearest Neighbor is the supervised machine learning algorithm can be used to solve both regression and classification problem. KNN does classification based on similarity identified in close proximity by calculating distance between two points.

The healthcare industry faces several challenges while adopting technology and has demanded changes [7] in electronic record management, computer diagnosis and data integration. Machine learning algorithms for classification has been widely adopted in the healthcare industry. This paper presents classification of orthopedic patients to the class-hernia, normal or spondylolisthesis based on data captured on biochemical features using machine learning algorithm. Different machine learning algorithms can be applied on same dataset, however, which ML algorithm will comparatively prove better depends upon the nature of the data [8] and preprocessing applied on the data. Here, three algorithms logistic regression, decision tree and K-NN are applied on the dataset and compared using confusion matrix.

2. RELATED WORKS AND BACKGROUND

2.1 SIMULATION APPROACHES

The simulation methods employed in image cytometry are primarily focused on generating suitable benchmarking data for image analysis algorithms. Their principal objective is to produce visually plausible synthetic data, mimicking the real acquired samples. In other words, the methods do not aim to precisely model the real-life biological processes, besides the visible phenomena observed in the eyepiece of optical microscope. The comprehensive tools for producing these datasets are often referred to as simulators or simulation frameworks.
The generated synthetic data is accompanied with inherent annotation, referred to as ground truth (GT), serving as a correct result, against which the image analysis algorithms are evaluated. The typical annotation, accompanying a synthetic sample, may include digital phantom, providing GT for deconvolution tasks, labeled image mask, serving as GT for segmentation, or cell lineage, serving as GT for tracking.

In following subsections, we will focus on various simulation scales, ranging from spots to tissues, as the scope of the models has to be often limited to a particular scale due to the computational constraints. We will also discuss the phases of the simulation, aiming to mimic the real-life acquisition process using an optical microscope. And finally, we will review various classes of computational models, which can be employed together in a simulator.

2.2 SIMULATION AT VARIOUS SCALES

As the microscopy and subsequent image analysis in the real world is done across a wide spectrum of scales and for a wide spectrum of possible applications, there is a need to model objects and structures at various scales. However, due to the high complexity of studied biological structures, and limits of contemporary computational and memory resources, the particular model often cannot.

be utilized for multiple scales, calling for different modeling strategies at each scale. Following list specifies various modeling scales, corresponding to the size of biological structures of interest, ranging from fine to coarse scale [5]:

i. spots and particles.

ii. subcellular components.

iii. cell nuclei.

iv. whole cell.

v. cell populations.

vi. tissues.

2.3 SIMULATION PHASES

Simulation workflow of contemporary simulation frameworks can be separated into three successive phases, mimicking the acquisition process of an optical microscope [5]:

1.digital phantom synthesis (synthetic specimen)

2.optical image formation (virtual microscope)

3.image acquisition (virtual acquisition device)

Although the term digital phantom is used in many fields with differ- Ent meanings, in the field of biomedical simulations it is a representation of structure and components of a specific object, such as nuclei, cell, or tissue in computer memory. Furthermore, the described object is not affected by any noise, blur, or uneven illumination. In a sense, the digital phantom represents the data we would be able to acquire from an ideal microscope with aberration-free optical system and perfect camera. Object described by a digital phantom is often composed of deformed or rotated geometrical shapes, which are filled with some procedural texture. Methods of synthesis of digital phantoms can be divided into parametric-based and learning-based, depending on the type of utilized modeling approach, as discussed in the following subsections.

Synthesized digital phantom, containing the structures of interest without any distortion or degradation, is subsequently advanced to a virtual optical microscope. A virtual optical microscope is modeling phenomena occurring in the real microscope, i.e., blurring process

occurring in the optical system, uneven illumination, and in the case of fluorescence imaging also virtual excitation-emission filters.

Finally, the simulation sequence is concluded by submitting the image to a virtual acquisition device. A virtual acquisition device models the phenomena manifesting themselves during image capture using digital image detectors, which is typically a CCD camera, CMOS camera, or a PMT detector. Among the modeled phenomena is dark current, resampling, fixed pattern noise, quantification uncertainty, amplification, and A/D conversion.

2.4 PARAMETRIC-BASED MODELS

Development of parametric-based models typically calls for deep bi- logical understanding of the object of interest. Gaining this prior knowledge often requires careful analysis of large datasets acquired from the real samples. The object properties are fully controlled by a set of user-defined parameters. However, the parameter space given by a particular parameter set is often vast, which makes these models difficult to tune for optimal configurations.

First parametric models employed in the ’90s were very limited by computational and memory resources available at the time. Due to these limitations, the models were based solely on basic shapes, such as spheres and ellipsoids. Furthermore, these models also completely omitted the texture modeling to save resources. The shape was the most important property, as it was essential for development of im- age analysis algorithms. Corresponding parametric model of a cell utilizing only basic shapes was proposed by Lockett et al. in 1998 [19]. The early attempts of modeling a texture for defining the internal object structure were done separately from the shape models. The first models based on procedural generation were proposed by Perlin in 1985 [20] and Timmer et al. in 1995 [21].

Subsequent models focused on simulation at various scales, from spots and particles presented by Cheezum et al. (2001) [22], Costes et al. (2004) [23] and Comeau et al. (2006) [24], to subcellular components presented by Gerencser et al. (2008) [25], cell nuclei presented by Svoboda et al. (2007) [26], and cell populations presented by Lehmus- sola et al. (2007) [27] and Malm et al. (2010) [28].

A simulation framework proposed by Ghaye et al. in 2012 [29] finally offered a comprehensive tool capable of generating synthetic objects of varying shape and texture. The added texture allowed the synthetic objects to approximate the real objects much more closely by incorporating

heterogeneity into the object structure. This made the segmentation of resulting samples much more complicated, offering a more comprehensive analysis of the developed algorithms. A few years later, a simulation framework proposed by Svoboda et al. in 2017 [30] implemented detailed simulation of developing cell population. The proposed framework employed complex parametric models of cell shape and texture, forming the digital phantom, and also included well-developed models of virtual microscope and acquisition device.

2.5 LEARNING-BASED MODELS

Learning-based models allow to delegate the arduous process of object analysis from human expert to a machine, which is able to automatically learn the important features from the given data. This approach alleviates the need for a large number of user-defined parameters and naturally offers higher variability in the generated data, as the machine is able to capture features that human would have difficulties to notice.

Training approaches utilized in development of learning-based models can be divided into three classes [5]:

i. principal component analysis.

ii. diffeomorphic deformations.

iii. deep generative models.

Among the first introduced learning-based approaches was the principal component analysis (PCA), which is a well-known method in statistics. The samples from the given training dataset are submitted to the analysis, producing feature vectors describing the shape of the objects. As the PCA concentrates the most important features to small number of vectors, it allows us to describe the shape in a compact and condensed manner by omitting the less important information. The utilization of PCA for shape modeling was proposed by Lehmussola et al. in 2008 [31].

At a later time, approaches based on diffeomorphic deformations surfaced. Here, the shape generation is performed utilizing an isomorphic transformation over the training samples, mapping one object shape to another such that both the transform function and its inverse are smooth. It other words, it is creating a function that defines a map- ping between shapes of all

objects. Models utilizing diffeomorphic deformations were proposed by Peng et al. in 2009 [32] a Simulation frameworks utilizing combination of PCA and diffeomorphic deformations to model the cell shape as well as the internal structure were proposed by Zhao et al. in 2007 [34] and Murphy in 2016 [35].

The newest class of learning-based approaches covers deep generative models, which are currently very popular among the scientific community [36, 6, 7, 8]. Generative models are able to capture the data distribution and generate new, never-before-seen samples of similar distribution. Furthermore, they are capable of working in an unsuper- vised manner, i.e., they do not require annotated data. Most common approaches employ models based on Variational Autoencoders (VAE), proposed by Kingma et al. in 2014 [12], and Generative Adversarial Networks (GAN), proposed by Goodfellow et al. in 2014 [11].and Xiong et al. in 2010 [33].

2.6 VARIATIONAL AUTOENCODERS

Variational Autoencoders, proposed by Kingma et al. in 2014 [12], are based on the original Autoencoder (AE), proposed by Hinton et al. in 1994 [37]. The main objective of Autoencoder is to represent (encode) the input data to a condensed form using dimensionality reduction. In other words, it creates a low-dimensional representation, referred to as code, of the high-dimensional input data. In this way, it works similarly to the principal component analysis, although the transformation itself is nonlinear. The model consists of two parts,

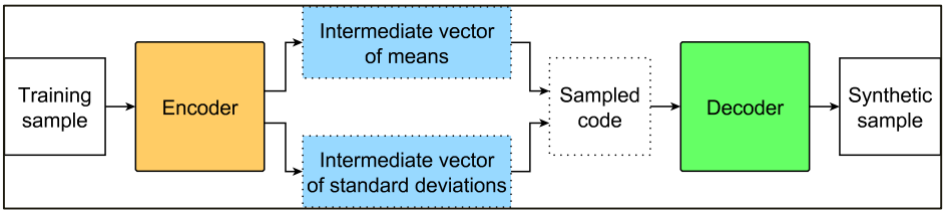

Figure 2.1: Conceptual diagram of the Variational Autoencoder. The encoder takes given training sample and generates the intermediate vectors. The code is then sampled from these vectors and decoded into synthetic sample using the decoder.

encoder and decoder. The encoder maps the input samples to a lower dimensional space with a constructive feature representation. This process can be repeated until the desired dimensional

space is reached. The decoder is then used to regenerate original feature space using reverse processing. The model was successfully applied for image denoising tasks and proposed as Denoising Autoencoder by Vincent et al. in 2008 [38]. However, the Autoencoder itself is not capable of generating new data.

The fundamental problem with Autoencoders is that the latent space of the coded samples is not continuous. Due to this limitation, decoder is only able to properly reconstruct the same samples that were used for training. This way, if we chose to decode random code that does not correspond to any training sample, the decoder output would not match the training distribution.

Furthermore, we also have to impose some limits on the values of vectors μ and σ to make all encoded code samples reasonably close to each other, which will allow for smooth interpolation and enable the generation of new plausible samples. In order to achieve this, the loss function combines a classic mean square error (MSE) with Kullback- Leibler (KL) divergence, denoted DKL, which evaluates how closely each latent variable Xi matches the normal distribution.

VAEs were originally proposed to overcome limitations of the original GANs, with amends oriented especially towards the mode collapse [39]. However, GAN models have already managed to take advantage of the important techniques employed in VAEs, such as divergence-based loss functions, considered in Wasserstein GAN (W- GAN) by Arjovsky et al. in 2017 [40]. Furthermore, combining both models into one, referred to as VAE-GAN, was also attempted in a few studies [41, 42, 15].

2.7 GENERATIVE ADVERSARIAL NETWORKS

Generative Adversarial Networks (GANs), proposed by Goodfellow et al. in 2014 [11], are a class of deep generative models employing machine learning. Their ability to capture important features directly from training data allows the generation of realistically looking images without the need of employing vast amounts of parameters or feature extraction.

During the training, two adversarial networks, generator and dis- criminator, compete in a non-cooperative game. The generator takes input noise vector and transforms it to a new (fake) sample, whereas.

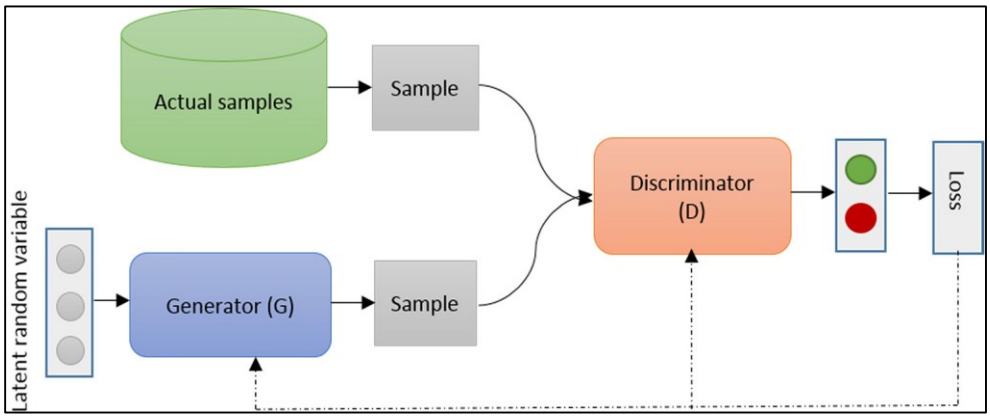

Figure 2.2: Conceptual diagram of the Generative Adversarial Network. The generator produces synthetic samples, given a random latent vector. The discriminator learns to distinguish between actual training samples and synthetic samples from the generator. The generator then learns from the loss provided by the discriminator. Source: Alom et al. (2019) [39]

the discriminator, behaving like a standard binary classifier, tries to distinguish between (real) samples from the training data and (fake) samples from the generator (see Figure 2.2). Furthermore, generator is newer given the actual training samples, it has available only the information provided by the discriminator. This training process continues until the generator is able to produce samples similar to the training data.

GANs are able to perform various tasks, such as:

i. image synthesis.

ii. image upscaling (also referred to as super-resolution).

iii. image colorization.

iv. image in painting.

v. image manipulation.

vi. text to image generation.

vii. image to image generation.

The model is able to properly learn only if the training dynamics between both networks is maintained so that neither of them is much stronger that the other, i.e., neither network wins the game. This makes the training very sensitive to model architecture and optimization parameters [43]. Specifically, the most common problems of GAN- based models are [44]:

non-convergence - During the training, the objective function oscillates and fails to converge toward the equilibrium.

mode collapse - The generator distribution collapses into single modality, resulting in samples of limited diversity.

vanishing gradient - The discriminator learns much faster than the generator and consequently provides very small gradient for it, hence, the generator learns very slowly or not at all.

overfitting - Unbalance between the generator and discriminator causes overfitting on the training data.

hyper parameter sensitivity - Only small portion of the hyper parameter space results in good convergence during training, the particular values are difficult to find.

The research on GANs has advanced substantially in recent years, proposing solutions to some of the suggested problems in subsequent variations of the model. The derived models are discussed in the following subsections.

2.8 ORIGINAL GAN

The original GAN model, proposed by Good fellow et al. in 2014 [11], utilized two fully connected neural networks, representing generator and discriminator. The minimax objective (value) function, defining loss for both networks, Deep Convolutional GAN
The architecture of Deep Convolutional GAN (DCGAN), proposed by Radford et al. in 2015 [14], substitutes the fully connected architecture of the original GAN for the fully convolutional architecture, utilizing convolutional neural networks, proposed by LeCun et al. in 1998 [45], without any fully connected layers in both the generator (see Figure 1.4) and the discriminator.

The adaptation of convolutional network architectures for GANs allowed for generation of images with much higher resolution. The learned convolutional kernels in every layer serve by design as feature extractors, capturing fine details and features present in the training.

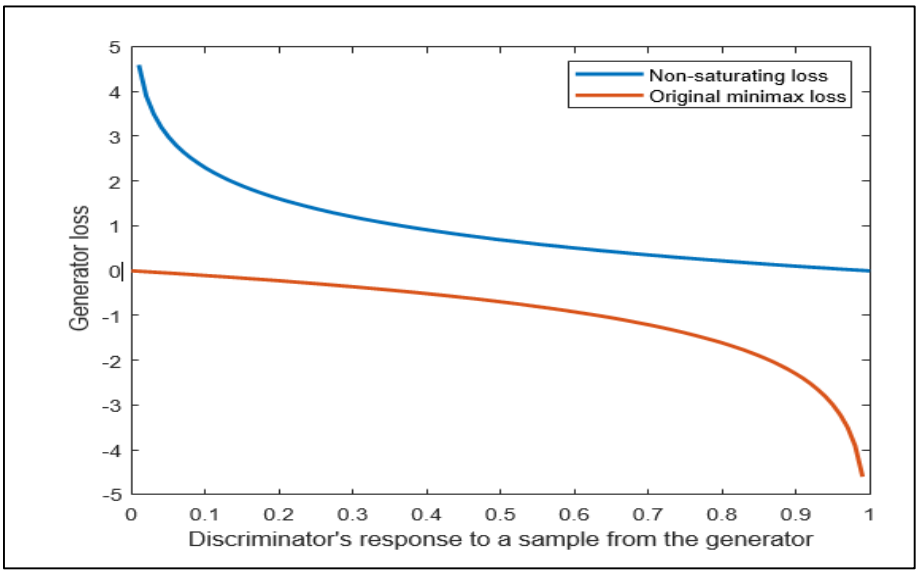

Figure 2.3: Comparison of the gradient given by the original minimax and the non-saturating loss functions of the generator.

The minimax loss gives low gradient when the sample is likely fake and high gradient when the sample is already close to the real data. On the other hand, the non-saturating loss gives high gradient for fake samples, allowing for efficient learning of the generator. data, which results in much higher fidelity of the generated outputs. The utilization of DCGANs in the field of biomedical image generation demonstrated their ability to generate realistic looking cellular specimens in 2D (CytoGAN), proposed by Goldsborough et al. in 2017 [13].

3D Shape Nets, proposed by Wu et al. in 2015 [46], adapted the convolutional architecture used previously for 2D images for the classification and retrieval of 3D shapes. The 3D convolutional architecture of Shape Nets was subsequently adapted for generative modeling in.

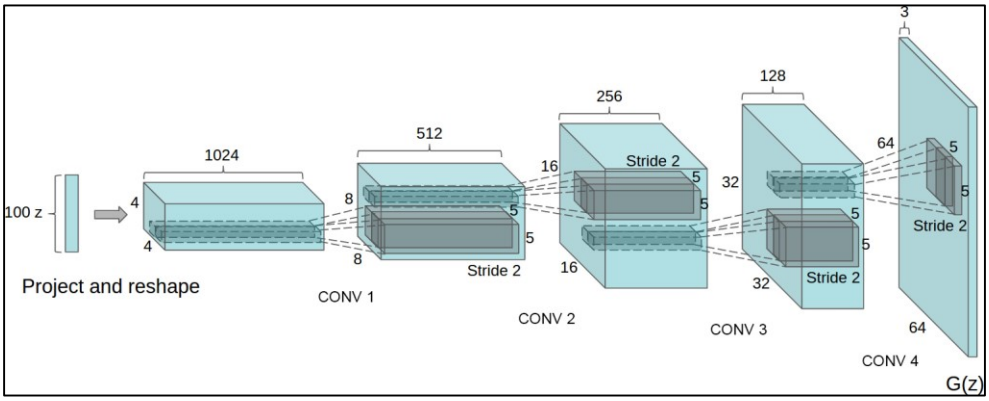

Figure 2.4: Fully convolutional generator architecture of DCGAN. The network does not utilize any fully-connected layers. Source: Radford et al. (2015) [14]

3D GAN, proposed by Wu et al. in 2016 [15], allowing for generation of fully 3D volumetric data, albeit thus far only in binary form.

2.9 CONDITIONAL GAN

The Conditional GAN (cGAN), proposed by Mirza et al. in 2014 [17], extends the original GAN model by including an additional conditional information for both the generator and the discriminator (see Figure 1.5). The conditioning is included as a supplementary input, giving prior auxiliary information to the model alongside the ordinary input data. The additional information may represent class labels or even data from other modalities. Consequently, this results in more stable training process, as it allows the model to learn depending on the given information.

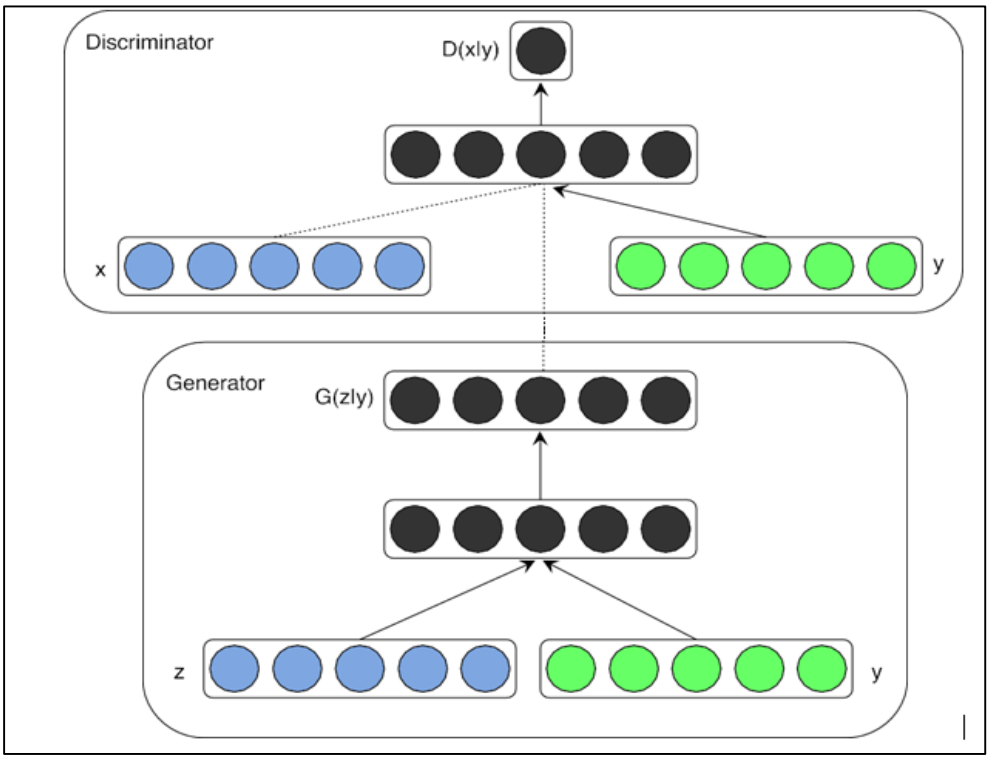

Figure 2.5: Conceptual diagram of the conditional GAN. The original GAN architecture is extended by additional input vector y, providing supplementary information for both the generator and the discriminator. Source: Mirza et al. (2014) [17]

for the conditioned generator G (z y).

The concept of cGANs was successfully used for conditional synthesis of images, e.g., handwritten digits [17]. Furthermore, it was especially important in works focusing on image-to-image translation, as suggested by Isola et al. in 2017 [47], and text-to-image synthesis, presented by Reed et al. in 2016 [48], and further reiterated by Bodnar in 2018 [49].

2.10 WASSERSTEIN GAN

The Wasserstein GAN (WGAN) model was proposed by Arjovsky et al. in 2017 [40], and further improved by Gulrajani et al. as Wasserstein GAN with gradient penalty (WGAN-GP) in 2017 [50]. The authors proposed new loss functions for the generator and the discriminator, producing better gradient for the generator, and solving the mode collapse problem of original

GANs. Furthermore, the generator loss was shown to be directly correlated with the quality of the generated samples, unlike the original minimax loss function.

Due to the different loss function, the authors also renamed discriminator to critic, as it no longer outputs the likelihood of a sample being real, but rather assigns an unbounded numerical score to it. During the training, the critic tries to maximize the difference between the score on real samples and the score on fake samples, and the generator tries to maximize the critic's output for fake samples.

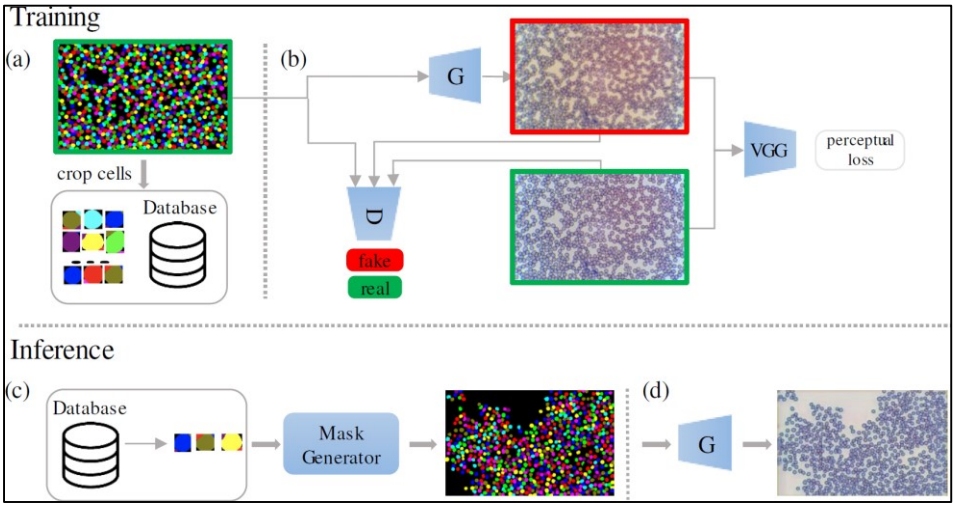

Figure 2.6: Synthesis of image and corresponding mask of the red blood cells. (a) All blood cell shape instances are extracted and saved to a database. (b) pix2pixHD framework is trained to translate segmentation mask to blood cell images. (c) Synthetic segmentation mask is created (inference stage). (d) Created segmentation mask is fed to the generator network to produce the blood cell image. Source: Bailo et al. (2019) [16]

2.11 GANS IN BIOMEDICAL IMAGE SYNTHESIS

In 2017 Goldsborough et al. [13] proposed CytoGAN for generation of cells in fluorescence microscopy. The approach utilized simple convolutional GAN (DCGAN) [14] for synthesis of RGB images of textured cells. However, the generated images were of low resolution and the whole image was synthesized in one pass. This way, the model has no explicit information about the cell shape and texture. Consequently, the network learns to produce an undesired noise present in the training data, and is not able to generate the corresponding cell mask, which is needed as an annotation.

The utilization of GANs to generate labels along with the textured tissue images was proposed by Han et al. in 2018 [10]. Most recent work, using conditional GANs (cGANs) [17] to synthesize blood cell images with corresponding masks (see Figure 1.6), was introduced by Bailo et al. in 2019 [16]. It employs pix2pixHD, proposed by Wang et al. in 2018 [51], capable of high-resolution image synthesis. The authors took advantage of existing pairs of blood cell images with corresponding segmentation mask and trained the pix2pixHD model to translate segmentation masks to blood cell images. Subsequently, they created a database of discrete blood cell shapes, which were cropped from the segmentation masks. By drawing random samples from this database, the authors generated new segmentation masks, which were used to synthesize corresponding blood cell images using the already trained pix2pixHD model. The presented results showed very good quality of the generated pairs, although the variability of the segmentation masks is limited by the size of the shape database. Furthermore, as the method generates images of high-resolution, it requires substantial computational resources.

With the increasing resolution of the synthesized images, the deep learning-based models have problems with quickly growing network complexity, and consequently, substantial computational and memory requirements. This is even more pronounced in case of adversarial training, where we have to train two networks of matching complexity simultaneously. An interesting approach was proposed by Wolterink et al. in 2018 [52] for blood vessel geometry synthesis. Instead of straight-forward generating of the blood vessel images, the authors generated a sequence of vectors of four parameters [x, y, z, r], where x, y, and z are Cartesian coordinates, and r is the radius of a blood vessel at these coordinates. The generated sequence is then converted to a 3D blood vessel object. The resulting model complexity for generating such 1D parametrized data is much lower than the complexity needed for a model producing the same vessels voxel by voxel in 3D.

2.12 DATA EVALUATION

As the simulated synthetic datasets with ground truth are intended as a substitution for real annotated datasets, and since the simulation models are always an approximation of reality [53], there is a natural desire to evaluate the plausibility of generated data. In other words, we want to investigate how similar are the synthetic data to the real data to guarantee the quality of the synthetic data for employment in validation of image analysis algorithms.

This topic is not the main focus of the thesis, consequently, the following paragraphs are intended purely as a brief overview of the available approaches. The evaluation of real and synthetic biomedical image data is the main research focus of Nečasová [54, 55].

As our models are focused on simulation of specimens in optical microscopy, we will concentrate on evaluation methods for images. The evaluation methods vary, depending on whether we want to compare pairs of data or whole datasets. Furthermore, the evaluation methods can be divided into two groups, qualitative and quantitative, depending on the general approach.

The qualitative approach is based on human visual comparison, utilizing the expert knowledge to assess the data. It constitutes of a blind test where the pairs of real and synthetic images are shown to an expert, who is asked to determine which is real and which is synthetic [56, 57]. The time an expert has to answer the query is also limited, in order to reduce deep analysis of particular details. Unfortunately, evaluating the images in this way is a time consuming and laborious task, which considerably limits the amount of data which can be processed. Furthermore, the results may differ greatly between various experts [3], making them difficult to reproduce.

On the other hand, the quantitative evaluation is a rigorous approach, producing exact, repeatable results. It takes advantage of data analysis tools commonly employed in statistics in order to obtain measures that are able to quantify the similarity of the data. The following measures are suitable for the pairwise assessment, where pairs of images are considered:

i. Mean Absolute Error (MAE).

ii. Peak Signal-to-Noise Ratio (PSNR).

iii. Structural Similarity Index (SSIM) - Wang et al. (2004) [58].

In the case of an evaluation on whole datasets, consisting of multitude of images, we can utilize the image descriptors to extract relevant characteristics from the data. Obtained image characteristics are then subject of the statistical analysis. The descriptors may represent information about color, shape, or texture of the objects of interest in the images.

3. MACHINE LEARNING

The idea of system learning, identifying patterns and making decision without human intervention can be described as machine learning [20]. It is a branch of artificial intelligence and depending on the learning approach can be divided into three categories as supervised, unsupervised and semi-supervised [21]. These categories are different mainly by the existence of labels (target values) for input data during the learning.

Several metrics are used to measure machine learning model performance:

i. Confusion matrix.
ii. Accuracy.
iii. F1-measure.
iv. Precision.
v. Recall

By these metrics, it is possible to compare all trained models with each other. As long as the problem is considered as classification problem, four basic metrics have to be introduced first, True vs. False and Positive vs. Negative output from the trained model. Binary problem with positive (1) and negative (0) classes can be taken as an example. A true positive (TP) output from models is the output that correctly predicts the positive class, and a true negative (TN) is the predicted correctly negative class. A false positive (FP) is an output from the model that incorrectly predicts the positive class, and a false negative (FN) is an incorrectly predicted negative class [22].

The first used metric, confusion matrix, takes as input at least two arrays, the predicted values for the data, and actual (original) values. As an output, it shows how many of predicted values by model belongs to categories TP, TN, FP or FN [23]. An example of a confusion matrix with two classes is shown in figure 3.1:

	Predicted class POSITIVE (spam ✉)	Predicted class NEGATIVE (normal 🕵)
Actual class POSITIVE (spam ✉)	TRUE POSITIVE (TP) ✉ ✉ 320	FALSE NEGATIVE (FN) ✉ 🕵 43
Actual class NEGATIVE (normal ✉)	FALSE POSITIVE (FP) ✉ ✉ 20	TRUE NEGATIVE (TN) ✉ 🕵 538

Figure 3.1: An example of the confusion matrix with two classes [24].

Second used metric is accuracy which represents the exact match of the predicted values and original values. If the predicted value of the i-th sample is y^i, the corresponding original value is yi, and the number of samples is n, then the accuracy can be defined as equation.

$$\text{Accuracy}(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} 1(\hat{y}_i = y_i)$$

Other used metrics are called precision, recall, and Fβ-measure. Precision represents the ability of the model to label just positive classes correctly. The recall represents the ability of the model to classify all positive classes. The harmonic mean of the precision and recall is called the Fβ-measure. Moreover, if β = 1, then the value is known as F1-measure and means that both recall and precision are equally

important [24]. The computation of these values in binary classification is shown in equations

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F_\beta = (1 + \beta^2) \frac{\text{precision} \times \text{recall}}{\beta^2 \text{precision} + \text{recall}}$$

Precision, recall, and Fβ-measure can be applied to each class independently in a multiclassification task. The detailed equations with necessary notation are explained in section Multiclass and multilabel classification of reference [24].

3.1 SUPERVISED MACHINE LEARNING METHODS

Supervised machine learning, also called learning from exemplars, provides learning on training data set with correct responses (targets). The algorithm generalizes to respond correctly to all possible inputs. So, at first, all methods from the supervised category need to have correct targets (labels) for training data, which can sometimes be challenging to obtain. If we had examples of every possible piece of input data, then we could put them to a big look-up table, so there would be no need to have machine learning at all. In reality, this is not a case [25].
However, the goal of the thesis is to compare the performance of the models. The chosen Python library scikit-learn supports the wide range of methods, and it is not possible to cover all of them in the range of this thesis. Instead, the focus was to select at least the most essential methods from a different perspective like Linear models, Support vector machines, and so on.

3.1.1 Logistic regression

Logistic regression is a machine learning classification algorithm that is utilized to predict the probability of a categorical dependent variable. Logistic regression can be reached by applying sigmoid function [26] on Linear regression. Equation (.5) represents sigmoid function whereas (.6) represents the Linear regression function with vector $w = (w_0, ..., w_p)$. Vector w represents coefficients of model. The w_0 represents the bias or intercept and the $(w_1, ..., w_p)$ represents the weight of the features $x = (x_1, ..., x_p)$. The result of sigmoid function is value between 0 and 1. Logistic regression can be then represented by equation (.7) together with mapping in equation (.8). Logistic regression will predict class 1, if the output of sigmoid function is more than 0.5. Otherwise, class 0 will be predicted [27].

One of the most suitable model for classification of anomaly detection is Logistic regression. Logistic regression is an algorithm based on dependent variables. It is one of the most popular and most widely used algorithms today. The term regression appears in this name, but Logistic regression is a classification algorithm. One of the possible implementation is realized by Python library scikit-learn, in class Logistic Regression [28]. The class offers a lot of parameters to be set while creating the instance. By the parameter with the name penalty, it supports optional regularization $\mathbf{1}$ which minimizes the cost function visible in equation (.9).

Similarly, the equation (.10) is solved as minimization of regularization $\mathbf{1}_2$ which is used by default. The last supported regularization is Elastic-Net which is combination of $\mathbf{1}_1$ and $\mathbf{1}_2$.

By parameter with the name solver, the class LogisticRegression offers to use different solvers. Implemented ones are liblinear, newton-cg, lbfgs, sag and saga. The solvers can be also used for multi classification problems by setting multiclass to 'multinomial'. The acronym lbfgs is used by an optimization algorithm the Broyden–Fletcher–Goldfarb–Shanno [29] from the family of quasi- Newton methods [30] with limited memory. It is recommended for small datasets due to performance issues.

In the real world, the data is rarely separable by the linear com- bination of features. It is possible to transform features into polynomial combinations to achieve better performance for models because it can better fit the data. Feature transformation into polynomial can be simply explained with concrete example of two features.

$X = [x_1, x_2]$ and degree 2. The transformed features into polynomial are
$X = [x_1, x_2, x_1x_2, x^2, x^2]$.

3.1.2 Support Vector Classifier

Support vector classifier (SVC) is the supervised machine learning method used for classification. It is one from the set of Support vector machines (SVMs) implemented also in the scikit-learn library.

SVMs were introduced by Vapnik in 1992 and became popular primarily because of significantly better classification performance on reasonably sized datasets. As long as they involve a data matrix inversion, which is an expensive operation for computation, they do not work well on extremely large datasets. The algorithm constructs a hyper-plane or set of them that separate the dataset into classes. In two dimensions and two classes, the hyper-plane is simply a line that creates a decision boundary. The distance between the nearest point to the decision boundary is called margin. The data points in each class that lies closest to the decision boundary are called support vectors. The best hyper-plane for SVM is the one with the maximum margin for data points in support vector [24].

SVMs covers a set of methods not only for classification but also for regression and outlier's detection. A subset of training points in the decision function, generally called

support vectors, cause the memory efficiency of these methods. SVMs support different kernel functions for the decision function. A custom kernel function is also possible to be used.

Class SVC [31] is concrete implementation of SVMs by scikit-learn library. By the kernel function, adjustable by parameter with the name kernel of the class SVC, a different decision boundary will be found by the method, for example, linear line or polynomial one of the features. Provided kernel functions [32] by SVMs in scikit-learn:

i. Linear.

ii. Polynomial.

where degree of polynomial function is specified by parameter with the name degree of class SVC.

Radial Basis Function - parameter with the name gamma of class SVC has to be specified. Gamma can be explained as the spread of the kernel (decision region). When the gamma is larger, then the data points have to be closer to each other to be affected. Library scikit-learn offer to set it to the 'scale' defined by equation (.11) or the 'auto' defined by equation (.12).

Sigmoid - also known as Hyperbolic Tangent Kernel [33] that uses the sigmoid function as a kernel.

The SVC method, similarly to the other supervised ones, takes as an input of fit method the array X holding the training samples and an array Y holding the labels. In our case, the array y contains just the values of the two classes.

3.1.3 Decision trees, ensemble methods and Random forest classifier

Decision trees (DTs): DTs are a non-parametric supervised learning method used for classification and regression. A model that predicts the value of a target variable is created by learning simple decision rules inferred from the data features [23]. DTs are easy to under- stand, interpret, and visualize. The cost if predicting data by decision tree is logarithmic in the number of data records used to train the tree. One of the most

important disadvantages of DTs includes the significant variance between the trees, which can be the result of small variations in the data. The solution is to use one of the ensemble methods that will be described later. Another problem can be overfitting that can be solved by setting the maximum depth of the tree.

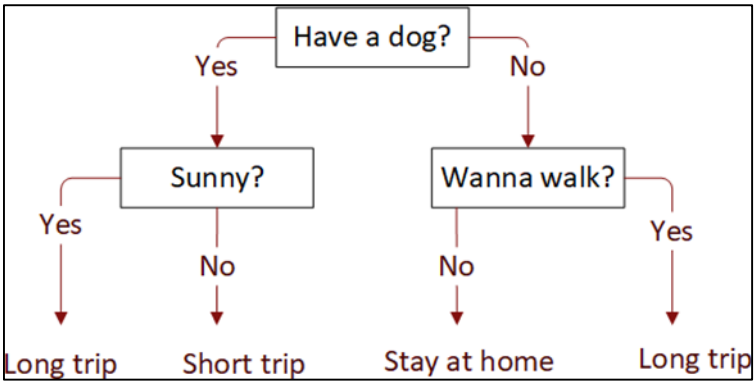

Figure 3.2: an example of the decision tree.

Ensemble Methods**:** The group of ensemble methods tries to com- bine the predictions of several machine learning techniques in order to improve the robustness of single prediction [23].

Most of the literature distinguishes between two popular groups of ensemble methods: boosting and bagging. Boosting is based on a combination of several very poor learners (estimators) to produce a powerful one. Bagging is the simpler ensemble method based on the aggregation of several predictions and picking the average of them [24].

The Random forest algorithm is one of bagging algorithms based on randomized decision trees that library scikit-learn learn includes. The Random forest algorithm is perturb-and-combine techniques [33] specifically designed for trees. By introducing randomness in the classifier, construction is created a diverse set of classifiers. The ensemble of individual predictions is combined into the average pre- diction of them.

However, the Random forest classifier is a concrete supervised implementation of bagging algorithms used for training in this thesis.

The algorithm is usable by class Random Forest Classifier [34] in the Python library scikit-learn. By parameter with the name n estimators of constructor, it is possible to set the number of decision trees that will be created in an ensemble from the training set. Classifier has to be also fitted with a sparse or dense array X holding the training samples and an array Y holding the

labels. During the construction of each tree the best split of each node is found either from all input features or a random subset of size max features from constructor's parameter. The reason is to decrease the variance of the forest estimators. The minimum number of samples required to split an internal node is defined by parameter with the name min samples split. A decision tree recursively partitions the internal node [35] until the maximum allowable depth is reached. Let the data at node m be rep- resented by Q. Each candidate split $\theta = (j, t_m)$ consists of a feature j and threshold t_m. The split of internal node data Q into two sub- sets $Q_{le\,f\,t}(\theta)$ and Q_{right} (θ) is defined in equations (.13) and (.14) respectively.

3.1.4 Naive Bayes - Multinomial Naive Bayes classifier

Naive Bayes methods are a group of supervised methods based on probability. In particular, they assume that every pair of features, given the value of the class variable, are conditionally independent. Under the hood, it means that the probability of getting the string of feature values (equation (.15)) is equal to the product of multiplying all of the individual probabilities (equation (.16)).

This assumption means that the features are independent of each other, which in reality is not true. However, this simplification is great for computation performance, and the algorithm itself shows results comparable to the classification methods in a certain domain. So, the classification rule of Naive Bayes' classifier is to select class Ci for which the computation from equation (.17) is the maximum [24].

Naive Bayes classifier is faster compared to others, more sophisticated methods because conditional feature distribution can be estimated independently as a one-dimensional distribution. When the number of dimensions' increases, much more data are needed, and this approach helps. Regarding the distribution of P ($X^k = a_k \mid Ci$),
the different naive Bayes' classifiers differ due to assumptions that they make [23]. Class Multinomial [36] is concrete implementation from Python scikit-learn library for multinomial distributed data. It was chosen because it is usually used in text classification [23].

3.2 UNSUPERVISED MACHINE LEARNING METHODS

In many circumstances, it is difficult to obtain labels for the data. For example, it could involve somebody to label the data manually. In this case, unsupervised machine learning methods are a good choice because they do not need labels at all. The algorithm has to recognize some similarities between input data points due to missing correct outputs. Identification of similarity between data points can lead to the classification too [24].

Unsupervised methods, in comparison with supervised one, cannot use external error criterion (the difference between targets and correct outputs) to learn on. They work based on the observation that an abnormal instance usually manifests as an outlier point that is distant from other instances. So the inputs that are closer together are recognized as similar (they are clustered together to the same class). In contrast, the inputs that are far apart are not recognized as similar (they are put into different classes if possible) [24].

3.2.1 K-means

K-means, one of the clustering algorithms [37], divides input data X into k disjoint clusters C of equal variance, minimizing a criterion known as the inertia, or within-cluster sum of squares that is defined by equation (.18).

Where μj describes the mean (usually called as the centroid) of the samples in the cluster and xi X. Means are not values from the input, but they are from the same space [5].

Concrete implementation of algorithm is offered by library scikit-learn as a class K-Means [38]. By parameter with the name n-clusters are possible to adjust numbers of clusters k. During the K-means algorithm, there are three steps. At first, the centroids of all clusters have to be initialized. The simplest way is to choose them randomly, but K-means from scikit-learn library offers a smarter way for their initialization. The basic idea is to set the values of centroids to be distant from each other, which leads to provably better results [39]. Then, the second step covers the assignment of all data points from X to the nearest centroid of clusters. To achieve minimal value for equation (18), all centroids are updated to the mean value of all the data points from X to each previous centroid. The difference between the new and old values of centroids is computed. Loop between the second and the third steps continues until the difference between the new and old values of centroids is less than the defined threshold.

The distance of each data point from X to centroids is often computed as Euclidean distance, but there are other alternatives also [24].

In general, the algorithm will always converge, but it may converge to a local minimum, not the global one. It strongly depends on the initial value of the centroids. The class K-Means offers to adjust the number of time the algorithm will run with the different centroid seeds by parameter with the name n-init. The disadvantage can be a long time until the algorithm finish. The algorithm K-means from library scikit-learn allows to be run in parallel, so it will run faster at the cost of memory.

3.2.2 Gaussian Mixture

A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters [23]. Gaussian distribution also known as Normal distribution, is a probability distribution that is symmetric about the mean. The data far from the mean are less frequent than the data near to mean. In figure 4.3 is an example of a probability density function where μ represents the mean.

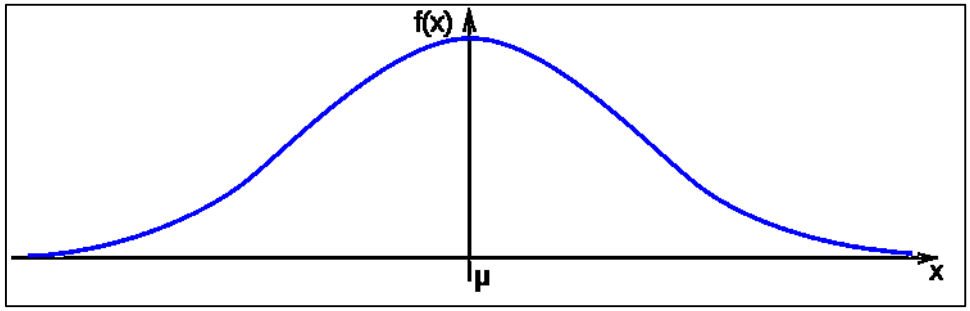

Figure 3.3: An example of a probability density function of Gaussian distribution.

For the fitting of the Gaussian mixture object created from a class Gaussian Mixture [37] offered by scikit-learn library, the expectation- maximization (EM) algorithm is used. EM algorithm gets around the problem to find out which points came from which latent component by an iterative process. In the first step, the EM algorithm assumes random components, randomly centered on data points or normally distributed around the origin. The number of components is possible to

adjust by parameter with the name n components of class Gaussian Mixture. The probability of being generated by each component for each point is computed. In the second step, one tweaks the parameters to maximize the likelihood of the data given those assignments. Loop between these steps continues, and the convergence is al- ways guaranteed to a local optimum [38]. The class Gaussian Mixture offers to adjust the number of time the algorithm will run with the different centers of components (means) by parameter with the name n init.
In the thesis, the classification of logs is discussed, so the number of components is already known. For log classification, three types of logs (three components) are needed. In general, selecting the number of components for the Gaussian mixture model can be more challenging. In this case, the (Bayesian information criteria) BIC criterion [39] can be used.

3.3 SEMI-SUPERVISED MACHINE LEARNING METHODS

Semi-Supervised methods are methods introduced due to two big disadvantages of supervised and unsupervised methods. The costly process of supervised learning on the big volume of the data and limited spectrum of application of unsupervised machine learning methods. Semi-supervised methods need a smaller amount of labeled data, and the rest are unlabeled. So typically, we use clustering (unsupervised method) to cluster bigger part of the data, and the rest we label with some supervised method [39].

3.4 MACHINE LEARNING PYTHON LIBRARIES

The programming language Python was chosen for all tools implemented in this thesis. This decision was made because the language is easy to use, freely available, there are many useful libraries for the usage area and it is becoming a default language for scientific computing [26]. As long as Python is a high-level scripting language, the tools don’t depend on the type of operating system. Most well-known machine learning libraries are written in Python:

Scikit-learn [40] is an open-source machine learning library. It provides a lot of models for supervised and unsupervised machine learning together with tools for model fitting, data pre-processing, and so on. It is the most suitable library for this thesis. This library is built on the top of numpy [41] (scientific computing) and scipy [42] (mathematics, science, and engineering) libraries. This library offers model persistence. So a trained model can be easily saved to a file

and load for data prediction anytime. There are some security and dependency limitations, but it is beneficial to save the trained model and use it when it is needed.

CNTK is a unified deep-learning toolkit from Microsoft Corporation. It is open-source since April 2015, and it provides us well-documented Python API [43].

Keras [44] is a high-level neural networks API. Library with a focus on fast experimentation. Keras is capable of running on the top of TensorFlow [45], CNTK, and Theano.

PyTorch [46] is an open-source machine learning framework that accelerates the path from research prototyping to production deployment.

4. EXPERIMENTAL RESULTS

4.1 DATA AND DATA PROCESSING

Biochemical features of orthopedic patient dataset [9] is used to classify patients into hernia, normal and spondylolisthesis, which are derived from the shape and orientation of the pelvis and lumbar spine. The dataset consists of total 310 rows and 7 columns pelvic incidence, pelvic tilt, lumbar lordosis angle, sacral slope, pelvic radius, degree spondylolisthesis, and class. All columns have numeric data except the column- class, which is string. The column class has three unique values- 'Hernia', 'Normal', 'Spondylolisthesis'. The dataset consists of 60 patients that belong to class Hernia, 100 patients belong to class Normal and 150 patients belong to class Spondylolisthesis. In this dataset column class is the target variable and all other columns are the feature vectors. The dataset contains no null values and no duplicate values. The target variable is encoded from text to number such that 0-Hernia, 1-Normal and 2- Spondylolisthesis.

Figure 4.1 below shows correlation between all feature vectors in the dataset. Correlation is useful in feature selection. High positive correlation is observed between pelvic incidence and sacral slope, pelvic incidence and lumbar lordosis angle lumbar lordosis angle and sacral slope. Moderate positive correlation is observed between pelvic incidence and pelvic tilt, lumbar lordosis angle and degree spondylolisthesis, sacral slope and degree spondylolisthesis. Low negative correlation is observed between pelvic radius and sacral slope.

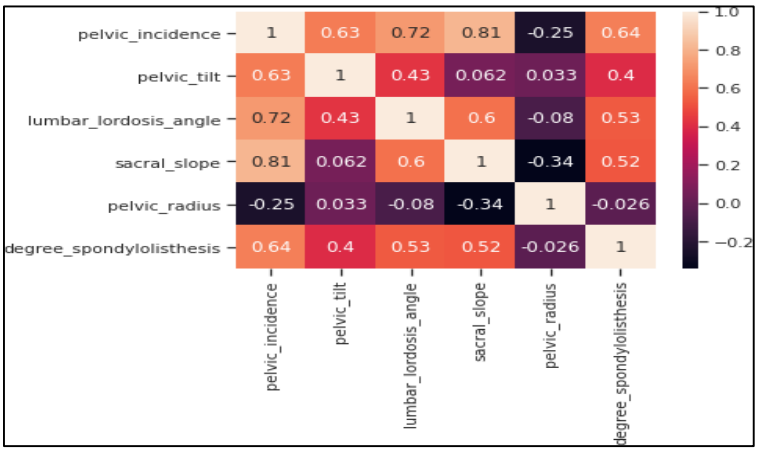

Figure 4.1: Correlation Between Feature Vectors

The dataset consists of three classes, so to understand the projection of clusters, Radviz plot [10]is plotted as shown in Figure 4.2. It is observed that almost all features in the dataset are equally important.

Feature scaling is done using standardization. The data is split in to training and testing in the ratio of 70:30. There are 217 samples in the training set and 93 samples in the test set.

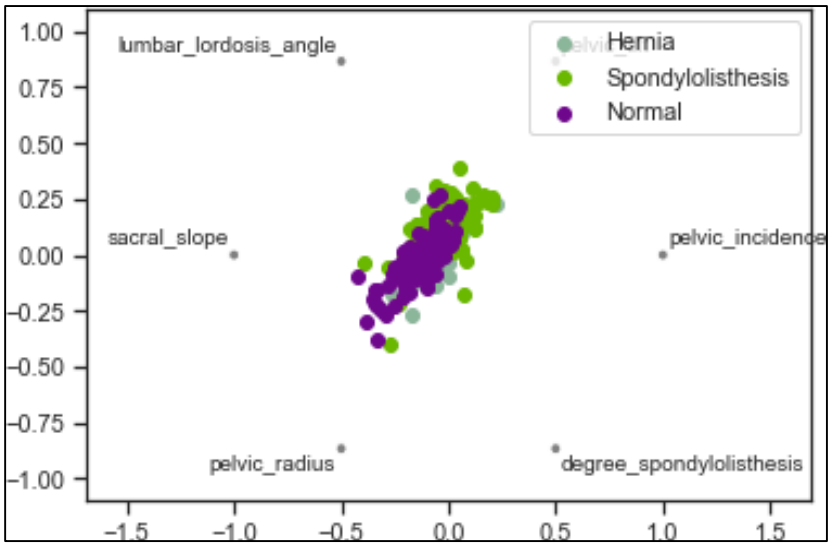

Figure 4.2: Radviz Plot.

4.2 CONFUSION MATRIX

To evaluate the performance of classification models confusion matrix [17] is used. It can be used for binary as well as multinomial classification. The target variable has positive and negative values represented in matrix as true positive (TP), true negative (TN), false positive (FP), false negative (FN). TP refers that the model predicted positive and actual was positive, in TN the model predicted negative and actual was negative, in FP the model predicted positive but actual was negative. FP is also known as type 1 error. In FN the model predicted negative but actual was positive. FN is also known as type 2 error.

As the dataset used here has three class so the confusion matrix is 3X3. In 3X3 confusion matrix the positive and negative are represented as shown in table 4.1.

4.3 RESULTS

In this section results obtained for confusion matrix and related indicators including accuracy score are presented for logistic regression, decision tree and KNN for comparative analysis.

Table 4.1. Confusion Matrix (Logistic Regression).

		Predicted		
		Hernia	Normal	Spondylolisthesis
Actual	Hernia	9	8	1
	Normal	6	16	2
	Spondylolisthesis	1	2	48

Table 4.2. Precision, Recall, F1-Score (Logistic Regression)

	Precision	Recall	F1-Score
Hernia	0.62	0.56	0.59
Normal	0.7	0.75	0.72
Spondylolisthesis	0.90	0.90	0.90

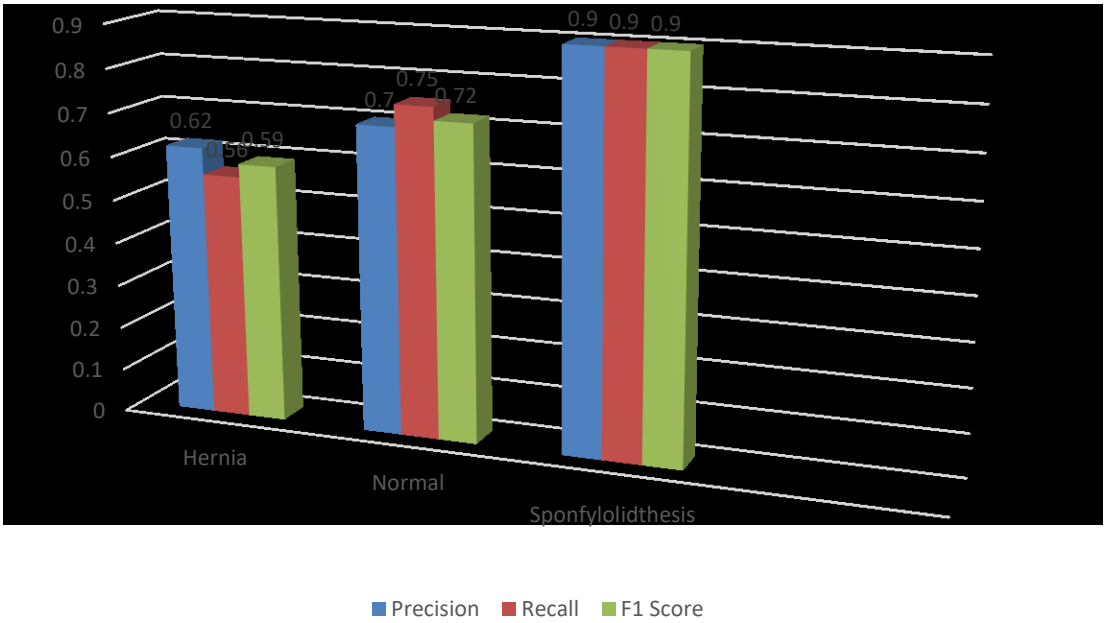

Figure 4.3: Precision, Recall, F1-Score (Logistic Regression).

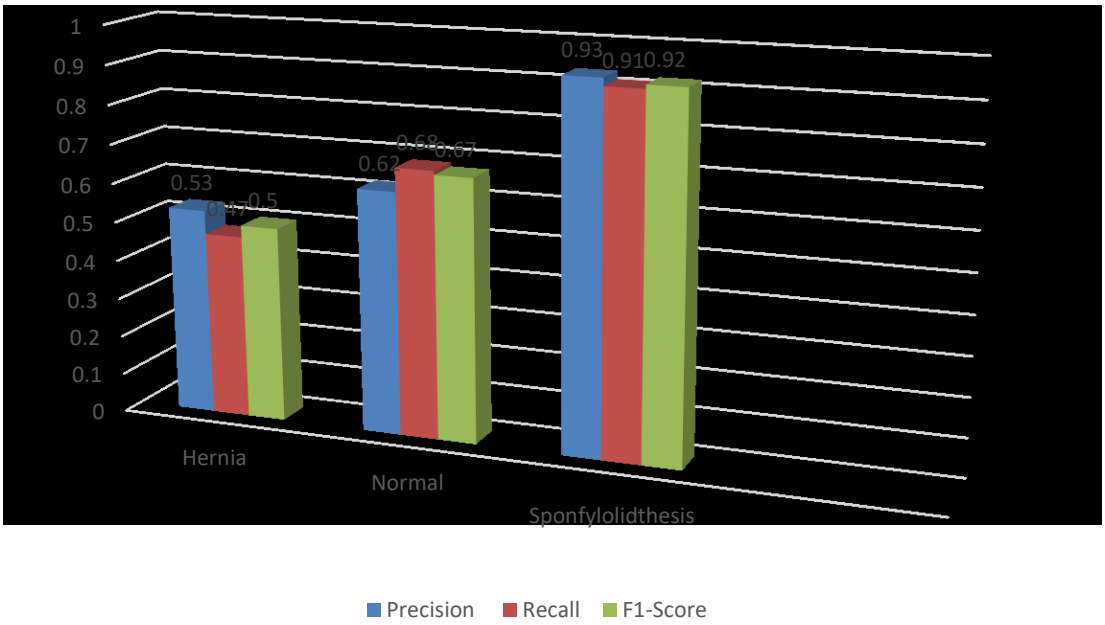

Figure 4.4: Precision, Recall, F1-Score (Decision Tree)

Table 4.3: Confusion Matrix (KNN)

		Predicted		
		Hernia	Normal	Spondylolisthesis
Actual	Hernia	9	8	1
	Normal	6	16	2
	Spondylolisthesis	1	2	48

Table 4.4: Precision, Recall, F1-Score (KNN)

	Precision	Recall	F1-Score
Hernia	0.58	0.52	0.55
Normal	0.64	0.69	0.66
Spondylolisthesis	0.92	0.92	0.92

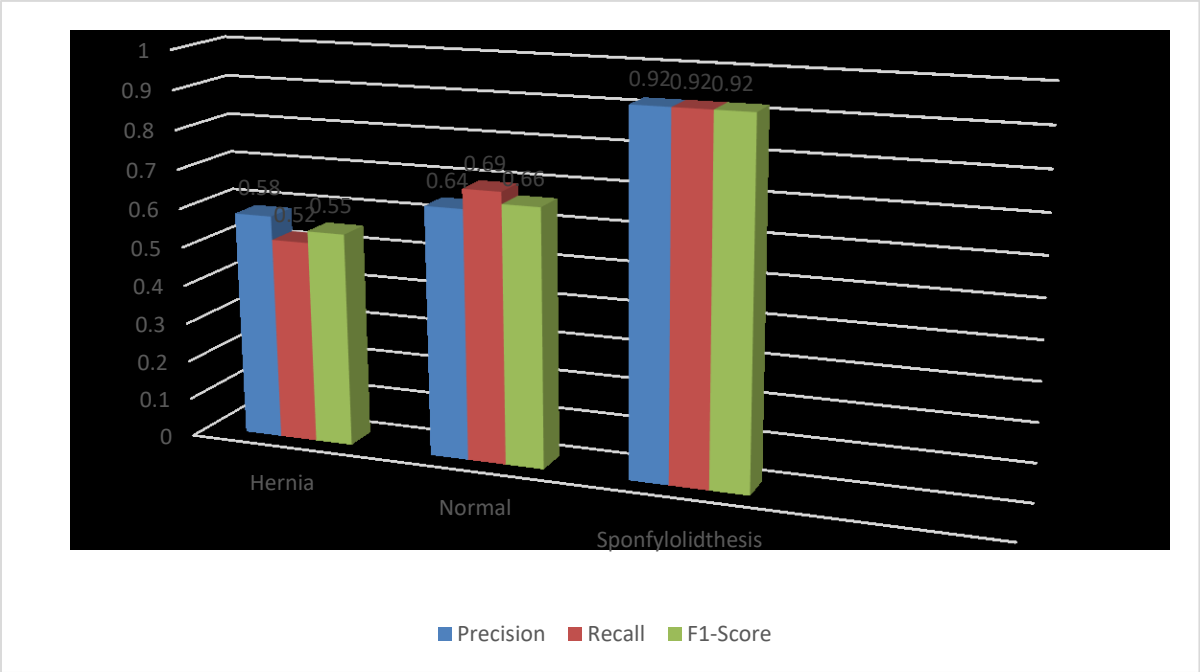

Figure 4.5: Precision, Recall, F1-Score (KNN)

As the machine learning algorithm is applied to medical data, errors in predicting all classes is equally important. The accuracy metric is therefore a useful indicator. The accuracy score of all the three, machine learning algorithm applied here to the data under study is as shown in table 4.6.

Table 4.5. Accuracy Score

Sr.on	Model	Accuracy Score
1	Logistic Regression	0.83
2	Decision Tree	0.77
3	KNN	0.79

5. CONCLUSION

Machine learning algorithms-logistic regression, decision tree and K-NN is used to train data captured on biochemical features of orthopedic patients to the classify it into three classes-hernia, normal or spondylolisthesis. Scaling of data is done using standardization before exposing it to machine learning algorithms. The data is split to train and test data in the ratio of 70:30. Confusion matrix and indicators-precision, recall, F1-score and accuracy are computed to measure performance of each of the three algorithms. Logistic regression uses parametric approach whereas decision tree and KNN uses non-parametric approach. There are total six features and all have continuous data, there are no categorical data in the feature vector and therefore we observe here, comparatively, decision tree therefore is not found suited for classification of this dataset. Also from the Radviz plot the preliminary insight gained is that the clusters are in close vicinity and therefore classification using K-NN has also comparatively not much accuracy to that of logistic regression. Thus, it is observed here that for the dataset the accuracy score of logistic regression is comparatively better than the decision tree and K-NN algorithm. Thus, it is concluded that performance of machine learning algorithm depends on the characteristic of the data.

REFERENCES

[1] Audrone Jakaitiene, Nonlinear Regression Models, in Encyclopedia of Bioinformatics and Computational Biology, 2019

[2] Münevver Köküer, Roger Green,et.l, Towards Automatic Risk Analysis for Hereditary Non-Polyposis Colorectal Cancer Based on Pedigree Data , Outcome Prediction in Cancer, 2007

[3] Joseph T. Hefner, Kandus C. Linde, Analytical Methods, in Atlas of Human Cranial Macromorphoscopic Traits, 2018

[4] T. Mitchell, "Decision Tree Learning", in T. Mitchell, Machine Learning, The McGraw-Hill Companies, Inc., 1997, pp. 52-78.

[5] P. Winston, "Learning by Building Identification Trees", in P. Winston, Artificial Intelligence, Addison-Wesley Publishing Company, 1992, pp. 423-442.

[6] M. Goldstein, Kn -nearest Neighbor Classification, Mathematics, Computer Science, IEEE Trans. Inf. Theory (1972), DOI:10.1109/TIT.1972.1054888, Corpus ID: 30697197

[7] Sumeet Dua, U. Rajendra Acharya, Prerna Dua, Machine Learning in Healthcare Informatics, DOI https://doi.org/10.1007/978-3-642-40017-9, Springer-Verlag Berlin Heidelberg 2014, Print ISBN978-3-642-40016-2

[8] M. Bichler, C. Kiss, A Comparison of Logistic Regression, k-Nearest Neighbor, and Decision Tree Induction for Campaign Management, Published in AMCIS 2004

[9] Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science

[10] Ankerst, M., Keim, D.A., Kriegel, H.-P.: Circle segments: A technique for visually exploring large multidimensional data sets. In: Visualization (1996)

[11] Andriy Burkov, The Hundred-Page Machine Learning Book, Publisher: Andriy Burkov (2019)

[12] "Building Decision Trees with the ID3 Algorithm", by: Andrew Colin, Dr. Dobbs Journal, June 1996

[13] "C4.5 Programs for Machine Learning", by J. Ross Quinlan, Morgan Kaufmann, 1993

[14] Breiman, Leo, et al. Classification and regression trees. CRC press, 1984.

[15] Kataria, A., & Singh, M. D. (2013). A Review of data classification Using K-nearest neighbour Algorithm. International Journal of Emerging Technology and Advanced Engineering, 3 (6), 354–360.

[16] Chomboon, K., Pasapichi, C., Pongsakorn, T., Kerdprasop, K., & Kerdprasop, N. (2015). An empirical study of distance metrics for k-nearest neighbor algorithm. In The 3rd International Conference on Industrial Application Engineering 2015 (pp. 280–285).

[17] Ting K.M. (2017) Confusion Matrix. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning and Data Mining. Springer, Boston, MA. https://doi.org/10.1007/978-1-4899-7687- 1_50

[18] Goutte C., Gaussier E. (2005) A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation. In: Losada D.E., Fernández-Luna J.M. (eds) Advances in Information Retrieval. ECIR 2005. Lecture Notes in Computer Science, vol 3408. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-31865-1_25

[19] LOCKETT, Stephen et al. Efficient, interactive, and three-dimen- sional segmentation of cell nuclei in thick tissue sections. *Cytom- etry: The Journal of the International Society for Analytical Cytology*. 1998, vol. 31, no. 4, pp. 275–286.

[20] PERLIN, Ken. An image synthesizer. *ACM Siggraph Computer Graphics*. 1985, vol. 19, no. 3, pp. 287–296.

[21] TIMMER, Jens et al. On generating power law noise. *Astronomy and Astrophysics*. 1995, vol. 300, pp. 707.

[22] CHEEZUM, Michael; WALKER, William; GUILFORD, William. Quantitative comparison of algorithms for tracking single fluo- rescent particles. *Biophysical journal*. 2001, vol. 81, no. 4, pp. 2378– 2388.

[23] COSTES, Sylvain; DAELEMANS, Dirk; CHO, Edward; DOBBIN, Zachary; PAVLAKIS,

George; LOCKETT, Stephen. Automatic and quantitative measurement of protein-protein colocalization in live cells. *Biophysical journal*. 2004, vol. 86, no. 6, pp. 3993–4003.

[24] COMEAU, Jonathan; COSTANTINO, Santiago; WISEMAN, Paul. A guide to accurate fluorescence microscopy colocalization mea- surements. *Biophysical journal*. 2006, vol. 91, no. 12, pp. 4611–4622.

[25] GERENCSER, Akos; NICHOLLS, David. Measurement of instan- taneous velocity vectors of organelle transport: mitochondrial transport and bioenergetics in hippocampal neurons. *Biophysical journal*. 2008, vol. 95, no. 6, pp. 3079–3099.

[26] SVOBODA, David et al. On simulating 3d fluorescent microscope images. In: *International Conference on Computer Analysis of Images and Patterns*. 2007, pp. 309–316.

[27] LEHMUSSOLA, Antti; RUUSUVUORI, Pekka; HUTTUNEN, Heikki; YLI-HARJA, Olli. Computational Framework for Simulating Flu- orescence Microscope Images with Cell Populations. *IEEE TMI*. 2007, vol. 26, no. 7, pp. 1010–1016.

[28] MALM, Patrik; BRUN, Anders; BENGTSSON, Ewert. Papsynth: simulated bright-field images of cervical smears. In: *Int. Sympo- sium on Biomedical Imaging: From Nano to Macro*. IEEE Press, 2010, pp. 117–120.

[29] GHAYE, Julien et al. Simulated biological cells for receptor count- ing in fluorescence imaging. *BioNanoScience*. 2012, vol. 2, no. 2, pp. 94–103.

[30] SVOBODA, D. et al. MitoGen: A Framework for Generating 3D Synthetic Time-Lapse Sequences of Cell Populations in Fluores- cence Microscopy. *IEEE Transactions on Medical Imaging*. 2017, vol. 36, no. 1, pp. 310–321. ISSN 0278-0062.

[31] LEHMUSSOLA, Antti et al. Synthetic images of high-throughput microscopy for validation of image analysis methods. *Proceedings of the IEEE*. 2008, vol. 96, no. 8, pp. 1348–1360.

[32] PENG, Tao et al. Instance-based generative biological shape mod- eling. In: *2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*. 2009, pp. 690–693.

[33] XIONG, Wei et al. Learning cell geometry models for cell image simulation: An unbiased approach. In: *2010 IEEE International Conference on Image Processing*. 2010, pp. 1897–1900.

[34] ZHAO, Ting et al. Automated learning of generative models for subcellular location: building blocks for systems biology. *Cytom- etry Part A*. 2007, vol. 71, no. 12, pp. 978–990.

[35] MURPHY, Robert. Building cell models and simulations from microscope images. *Methods*. 2016, vol. 96, pp. 33–39.

[36] TURHAN, Ceren Güzel; BILGE, Hasan Sakir. Recent Trends in Deep Generative Models: a Review. In: *2018 3rd International Conference on Computer Science and Engineering (UBMK)*. 2018, pp. 574–579.

[37] HINTON, Geoffrey et al. Autoencoders, minimum description length and Helmholtz free energy. In: *Advances in neural informa- tion processing systems*. 1994, pp. 3–10.

[38] VINCENT, Pascal et al. Extracting and composing robust fea- tures with denoising autoencoders. In: Proceedings of the 25th international conference on Machine learning. 2008, pp. 1096–1103.

[39] ALOM, Md Zahangir et al. A state-of-the-art survey on deep learning theory and architectures. Electronics. 2019, vol. 8, no. 3, pp. 292.

[40] ARJOVSKY, Martin; CHINTALA, Soumith; BOTTOU, Léon. Was- serstein Generative Adversarial Networks. In: Proceedings of the 34th International Conference on Machine Learning. International Convention Centre, Sydney, Australia: PMLR, 2017, vol. 70, pp. 214–223. Proceedings of Machine Learning Research.

[41] LARSEN, Anders Boesen Lindbo; SØNDERBY, Søren Kaae; LARO- CHELLE, Hugo; WINTHER, Ole. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*. 2015.

[42] LAMB, Alex; DUMOULIN, Vincent; COURVILLE, Aaron. Dis- criminative regularization for generative models. arXiv preprint arXiv:1602.03220. 2016.

[43] MESCHEDER, Lars; GEIGER, Andreas; NOWOZIN, Sebastian. Which Training Methods for GANs do actually Converge? In: International Conference on Machine learning (ICML). 2018.

[44] SALIMANS, Tim; GOODFELLOW, Ian; ZAREMBA, Wojciech; CHEUNG, Vicki; RADFORD, Alec; CHEN, Xi. Improved tech- niques for training GANs. In: Advances in neural information pro- cessing systems. 2016, pp. 2234–2242.

[45] LECUN, Yann; BOTTOU, Léon; BENGIO, Yoshua; HAFFNER, Patrick, et al. Gradient-based learning applied to document recog- nition. *Proceedings of the IEEE*. 1998, vol. 86, no. 11, pp. 2278–2324.

[46] WU, Zhirong; SONG, Shuran; KHOSLA, Aditya; YU, Fisher; ZHANG, Linguang; TANG, Xiaoou; XIAO, Jianxiong. 3D Shape- Nets: A deep representation for volumetric shapes. In: *Proceed- ings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1912–1920.

[47] ISOLA, Phillip; ZHU, Jun-Yan; ZHOU, Tinghui; EFROS, Alexei A. Image-to-image translation with conditional adversarial net- works. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[48] REED, Scott; AKATA, Zeynep; YAN, Xinchen; LOGESWARAN, Lajanugen; SCHIELE, Bernt; LEE, Honglak. Generative adver- sarial text to image synthesis. *arXiv preprint arXiv:1605.05396*. 2016.

[49] BODNAR, Cristian. Text to image synthesis using generative adversarial networks. *arXiv preprint arXiv:1805.00676*. 2018.

[50] GULRAJANI, Ishaan; AHMED, Faruk; ARJOVSKY, Martin; DU- MOULIN, Vincent; COURVILLE, Aaron. Improved Training of Wasserstein GANs. In: *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017, pp. 5767–5777.

[51] WANG, Ting-Chun; LIU, Ming-Yu; ZHU, Jun-Yan; TAO, Andrew; KAUTZ, Jan; CATANZARO, Bryan. High-resolution image syn- thesis and semantic manipulation with conditional gans. In: *Pro- ceedings of the IEEE conference on computer vision and pattern recog- nition*. 2018, pp. 8798–8807.

[52] WOLTERINK, Jelmer; LEINER, Tim; ISGUM, Ivana. Blood vessel geometry synthesis using generative adversarial networks. *arXiv preprint arXiv:1804.04381*. 2018.

[53] SARGENT, Robert. Verification and validation of simulation mod- els. In: Proceedings of the 2010 Winter Simulation Conference. 2010, pp. 166–183.

[54] NEČASOVÁ, Tereza. Statistical methods for comparison of real and synthetic biomedical image data [online]. 2019. Available also from: https://is.muni.cz/th/o70rv/.

[55] NEČASOVÁ, Tereza; SVOBODA, David. Visual and Quantitative Comparison of Real and Simulated Biomedical Image Data. In: Computer Vision – ECCV 2018 Workshops. Cham: Springer Inter- National Publishing, 2019, pp. 385–394. ISBN 978-3-030-11024-6.

[56] MEYER, Gary; RUSHMEIER, Holly; COHEN, Michael; GREEN- BERG, Donald; TORRANCE, Kenneth. An experimental evalua- tion of computer graphics imagery. *ACM Transactions on Graphics (TOG)*. 1986, vol. 5, no. 1, pp. 30–50.

[57] MALM, Patrik; BRUN, Anders; BENGTSSON, Ewert. Simulation of bright-field microscopy images depicting pap-smear specimen. *Cytometry Part A*. 2015, vol. 87, no. 3, pp. 212–226.

[58] WANG, Zhou; BOVIK, Alan; SHEIKH, Hamid; SIMONCELLI, Eero, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*. 2004, vol. 13, no. 4, pp. 600–612.