FEATURE LEARNING AS A TOOL TO IDENTIFY EXISTENCE OF
MULTIPLE BIOLOGICAL PATTERNS

by

Aleksandr Patsekin

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science

Department of Computer and Information Technology

West Lafayette, Indiana

May 2018
THE PURDUE UNIVERSITY GRADUATE SCHOOL

STATEMENT OF COMMITTEE APPROVAL

Prof. William G. McCartney, Co-Chair
Department of Computer and Information Technology

Prof. Joseph P. Robinson, Co-Chair
Department of Biomedical Engineering

Prof. Bartek Rajwa
Department of Basic Medical Sciences

Prof. John A. Springer
Department of Computer and Information Technology

Approved by:

Prof. Eric T. Matson
Head of the Graduate Program
ACKNOWLEDGMENTS

I am very grateful to my committee team members for supporting me throughout the whole process of earning this degree: Prof. William G. McCartney, Prof. Joseph P. Robinson, Prof. Bartek Rajwa, and Prof. John A. Springer. I would like to acknowledge Gretchen Lawler for formatting revision.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>viii</td>
</tr>
<tr>
<td>GLOSSARY</td>
<td>x</td>
</tr>
<tr>
<td>LIST OF ABBREVIATIONS</td>
<td>xi</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>xii</td>
</tr>
<tr>
<td>CHAPTER 1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Significance of the Problem</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Scope of the Study</td>
<td>3</td>
</tr>
<tr>
<td>1.3 Research Hypothesis</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Assumptions</td>
<td>4</td>
</tr>
<tr>
<td>1.5 Limitations</td>
<td>4</td>
</tr>
<tr>
<td>1.6 Delimitations</td>
<td>5</td>
</tr>
<tr>
<td>CHAPTER 2. REVIEW OF THE LITERATURE</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Methodology of the Literature Review Process</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Related Literature Review</td>
<td>7</td>
</tr>
<tr>
<td>2.3 Application to Biological Domain</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Autoencoders</td>
<td>13</td>
</tr>
<tr>
<td>2.5 Common Methods in Related Literature</td>
<td>14</td>
</tr>
<tr>
<td>2.6 Summary</td>
<td>15</td>
</tr>
<tr>
<td>CHAPTER 3. METHODOLOGY</td>
<td>16</td>
</tr>
<tr>
<td>3.1 Research Framework</td>
<td>16</td>
</tr>
<tr>
<td>3.1.1 Feature learning</td>
<td>17</td>
</tr>
</tbody>
</table>
3.1.1.1 Deep convolutional autoencoder ................................................................. 17
3.1.1.2 Pre-trained deep CNN feature extraction ..................................................... 20
3.1.1.3 Representation learning using vector quantization ........................................ 22
3.1.2 Subsequent clustering ..................................................................................... 24
  3.1.2.1 Parametric clustering .................................................................................. 24
  3.1.2.2 Non-parametric clustering ......................................................................... 24
3.2 Research Sampling Approach .......................................................................... 25
  3.2.1 Scatter patterns collected by bacteria rapid detection using optical scattering
technology (BARDOT) ......................................................................................... 25
  3.2.2 Stimulating peripheral activity to relieve conditions (SPARC) ....................... 26
3.3 Variables ............................................................................................................. 27
3.4 Assessment Instruments ..................................................................................... 27
  3.4.1 Clustering quality .......................................................................................... 27
    3.4.1.1 V-measure, homogeneity, completeness .................................................. 28
    3.4.1.2 Mean silhouette coefficient ..................................................................... 30
    3.4.1.3 Graphical aid for interpretation of clustering .......................................... 30
  3.4.2 SPARC dataset methodology ....................................................................... 31
CHAPTER 4. PRESENTATION OF DATA & FINDINGS ............................................... 32
4.1 BARDOT Setup and Findings ............................................................................. 32
  4.1.1 Convolutional autoencoder .......................................................................... 32
    4.1.1.1 Setup configuration .................................................................................. 32
    4.1.1.2 Presentation of the results ....................................................................... 34
  4.1.2 Pre-trained deep CNN feature extraction ...................................................... 38
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.2.1 Setup configuration</td>
<td>38</td>
</tr>
<tr>
<td>4.1.2.2 Presentation of the results</td>
<td>39</td>
</tr>
<tr>
<td>4.1.3 Representation learning using vector quantization</td>
<td>42</td>
</tr>
<tr>
<td>4.2 Setup and Findings for SPARC Dataset</td>
<td>43</td>
</tr>
<tr>
<td>CHAPTER 5. CONCLUSIONS AND DISCUSSION</td>
<td>46</td>
</tr>
<tr>
<td>APPENDIX A. COMPUTATIONAL GRAPH FOR CONVOLUTIONAL AUTOENCODER</td>
<td>50</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>51</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 4.1. Convolutional layer configuration of autoencoder for BARDOT dataset .................. 33
Table 4.2. Hyper-parameter setups of autoencoder for BARDOT dataset ................................ 33
Table 4.3. Cross-validated clustering quality metrics for autoencoder on BARDOT dataset A . . 35
Table 4.4. Cross-validated clustering quality metrics for autoencoder on BARDOT dataset B . . 36
Table 4.5. Cluster-number estimation based on extracted features using autoencoder .......... 38
Table 4.6. Convolutional layers configuration of pre-trained CNN for BARDOT dataset ......... 39
Table 4.7. Hyper-parameter setups of pre-trained CNN for BARDOT dataset ......................... 39
Table 4.8. Cross-validated clustering quality metrics achieved on BARDOT dataset A using features extracted with CNN classifier ............................................................... 40
Table 4.9. Cross-validated clustering quality metrics achieved on BARDOT dataset B using features extracted with pre-trained CNN. ................................................................. 41
Table 4.10. Cross-validated cluster-number estimation based on extracted features with pre-trained CNN ................................................................. 41
Table 4.11. Hyper-parameters for representation learning using vector quantization .............. 42
Table 4.12. Cross-validated clustering quality metrics achieved on BARDOT datasets using features extracted with representation learning using vector quantization. ............... 42
Table 4.13. Cross-validated cluster-number estimation based on extracted features with representation learning using vector quantization. ........................................ 42
Table 4.14. Hyper-parameter setups of convolutional autoencoder for SPARC dataset ............ 43
Table 5.1. Consolidated, cross-validated results of clusters quality for all three proposed methods for BARDOT datasets .................................................................................. 46
LIST OF FIGURES

Figure 3.1. Overall pipeline of the experiment, which includes feature learning and subsequent clustering......................................................... 17

Figure 3.2. Deep convolutional autoencoder. Contains two phases: feature learning and clustering experiment data............................................................. 20

Figure 3.3. Deep CNN pipeline. Demonstrates pipeline for supervised CNN training................... 21

Figure 3.4. Feature extraction model using trained deep CNN................................................................. 22

Figure 3.5. Pipeline of the vector quantization learning model. Feature learning in the top row; inference in the middle row. .............................................................. 23

Figure 3.6. Structure of two BARDOT datasets; (a) dataset A employed for training and testing comprises five classes; (b) dataset B employed for testing comprises five different classes. ................................................................. 26

Figure 3.7. Cross-validation scheme to validate models accuracy................................................................. 27

Figure 4.1. Loss function curves against training step........................................................................... 34

Figure 4.2. Restoration capabilities on BARDOT dataset of AE with simple max-pooling: (a) input, (b) output. AE with max-pooling and argmax: (c) input, (d) output. ......................... 35

Figure 4.3. Scatter plot of labeled feature vectors obtained from BARDOT dataset A with t-SNE dimensionality reduction................................................................. 36

Figure 4.4. Autoencoder results on BARDOT dataset B: (a) ground truth with t-SNE embedding, (b) k-means clustering with K=5 and t-SNE embedding.......................... 37

Figure 4.5. T-SNE embedding of feature-vectors obtained on BARDOT dataset A with convolutional autoencoder; (a) ground truth for five classes; (b) DBSCAN clustering with five clusters. Red dashes demonstrate outliers for cluster #4 and #5. .............. 38
Figure 4.6. CNN classifier performance metric during training process: (a) loss function, (b) accuracy of classification. Smoothing coefficient = 0.992. ........................................ 40

Figure 4.7. T-SNE embedding of feature vectors obtained through vector quantization on BARDOT dataset A. (a) ground truth for five classes; (b) HDBSCAN clustering with five clusters; red circles highlight mismatch in clustering. ........................................ 43

Figure 4.8. Confusion matrix obtained from clustering SPARC dataset on a per-image basis. Results are validated using Monte Carlo cross-validation. ........................................ 44

Figure 4.9. Application of autoencoder with k-means clustering (silhouette-based estimation of K parameter) on SPARC dataset. ............................................................................. 45

Figure 5.1. Boxplots for distributions over cross-validation runs for (1) vector quantization on dataset A, (2) autoencoder on dataset A, (3) vector quantization on dataset B, (4) autoencoder on dataset B. ................................................................. 47

Figure 5.2. Zoomed region of PCA 3-D embedding of features extracted from BARDOT dataset B, using CNN classifier trained on BARDOT dataset A with a different set of classes. Colors of images represent ground truth classes. Red arrows point to visually different bacteria that were positioned close to *Salmonella anatum* (violet-colored) class. PCA embedding describes 95.6% of variance in 200-D vector space. ............................. 48
GLOSSARY

Artificial Neural Network – a computer system usually represented by a pipeline of many trainable parameters connected by mathematical operators and functions.

Machine Learning – a technique to solve problems through learning from provided datasets.

Python – a high-level programming language that has different built-in modules for solving computational research problems.

Tensorflow – a software library typically used for machine learning applications.
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argmax</td>
<td>The arguments of the maxima function</td>
</tr>
<tr>
<td>BARDOT</td>
<td>Bacteria Rapid Detection using Optical scattering Technology</td>
</tr>
<tr>
<td>CNN</td>
<td>Convolutional Neural Network</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphical Processing Unit</td>
</tr>
<tr>
<td>MLP</td>
<td>Multilayer Perceptron</td>
</tr>
<tr>
<td>ReLU</td>
<td>Rectified Linear Unit</td>
</tr>
<tr>
<td>SGD</td>
<td>Stochastic Gradient Decent</td>
</tr>
<tr>
<td>SPARC</td>
<td>Stimulating Peripheral Activity to Relieve Conditions</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
</tbody>
</table>
ABSTRACT

Author: Patsekin, Aleksandr, V. MS
Institution: Purdue University
Degree Received: May 2018
Title: Feature Learning as a Tool to Identify Existence of Multiple Biological Patterns.
Major Professor: William G. McCartney

This paper introduces a novel approach for assessing multiple patterns in biological imaging datasets. The developed tool should be able to provide most probable structure of a dataset of images that consists of biological patterns not encountered during the model training process. The tool includes two major parts: (1) feature learning and extraction pipeline and (2) subsequent clustering with estimation of number of classes. The feature-learning part includes two deep-learning techniques and a feature quantitation pipeline as a benchmark method. Clustering includes three non-parametric methods. K-means clustering is employed for validation and hypothesis testing by comparing results with provided ground truth. The most appropriate methods and hyper-parameters were suggested to achieve maximum clustering quality. A convolutional autoencoder demonstrated the most stable and robust results: entropy-based V-measure metric 0.9759 on a dataset of classes employed for training and 0.9553 on a dataset of completely novel classes.
CHAPTER 1. INTRODUCTION

Classification and pattern-recognition problems considered extremely difficult for early artificial intelligence (AI) implementations 20 years ago are now routinely solved by modern AI technologies such as deep learning with unprecedented accuracy.

In biological image analysis, machine learning models are often utilized for such tasks as classification, segmentation or target detection (Carneiro, Zheng, Xing, & Yang, 2017). A traditional approach in this domain includes pre-processing of the dataset by experts in the subject matter. Then follows the feature extraction process with the subsequent application of machine learning algorithms, such as support vector machine (SVM). The described approach has obvious advantages, such as low computational power usage and fast inference. However, since 2012 different machine learning techniques have been widely applied, demonstrating promising results. The term "deep learning" describes the design and training of artificial neural network containing many hidden layers (Goodfellow, Bengio, & Courville, 2016, p. 1). The number of layers depends on the complexity and abstraction of the classification problem. Application of deep learning to the biological domain is an active area of research. Using deep trained classifiers, it is possible to achieve state-of-the-art results in a variety of biological imaging areas. Despite of this fact, significant limitations are inherent in this method.

1.1 Significance of the Problem

With respect to deep-learning classifiers, there is a strict requirement for a prior knowledge of class numbers. Another limitation of traditionally employed machine learning is reliance on relatively large training datasets. Although these requirements may be easy to fulfill for classification of natural images, it is often impossible to meet technical expectations for biomedical
patterns or images; obtaining labeled data for datasets of medical images is a very challenging task and manual annotation is laborious, expensive, and often ambiguous (Dundar, Kou, Zhang, He, & Rajwa, 2015; Sommer, Hoefler, Samwer, & Gerlich, 2017; Xu et al., 2015).

Researchers from Purdue University presented a rapid technology for obtaining preliminary results for identification of bacterial colonies on agar plates based on light-scatter patterns (Bayraktar et al., 2006). In their work, they emphasized the importance of rapid prescreening methods when it comes to food poisoning or bioterrorism prevention. However, one possible flaw of this technology is the stage of classification of the obtained images. This research paper described Zernike moments as a feature extracting technique. According to the results reported, only 84% of patterns were correctly classified. Further progress in this technology was described in a paper by Dundar, Kou, Zhang, He, and Rajwa (2015). Different classification models were applied to compare their accuracy. Surprisingly, feature-learning method based on vector quantization demonstrated nearly perfect classification accuracy – 97.89% on a dataset of four bacteria classes (Coates, Ng, & Lee, 2011). As a follow-up study, they proposed application of deep-learning models with large-scale datasets to see if performance improves. However, one may notice that all these learning techniques are suitable only for end-to-end classification with models trained in a supervised manner. From a practical point of view, in its current state this technology is not capable of identifying novel strains of bacteria owing to the nature of supervised classification models (Sommer et al., 2017). A serious outbreak of foodborne illness took place in Germany in 2011, caused by a novel strain of *Escherichia coli* O104:H4 bacteria (Frank et al., 2011). There were 4075 cases reported with 48 deaths (Cui, Li, & Yang, 2013). To prevent such catastrophes, technologies like scatter-pattern analysis should be able to identify novelty in datasets. The research conducted here is intended to employ unsupervised and pre-trained feature
learning and subsequent clustering techniques to address these issues. The efficiency of deep learning as a feature extraction method in the biological area has been demonstrated by many researchers in recent years (Kraus et al., 2017; Li, Zeng, Peng, & Ji, 2017; Wang, MacKenzie, Ramachandran, & Chen, 2015). When it comes to novelty detection in an unsupervised manner, successful application of deep learning was described in a paper by Sommer et al., in which they applied this method to cellular phenotype detection in genome-scale screening data (2017).

1.2 Scope of the Study

In this research project, we propose to examine the problem of biological pattern recognition under three simultaneous constraints: (1) highly limited training datasets, (2) lack of knowledge regarding the features defining the putative patterns, and (3) an undetermined number of classes. The long-term goal of the project is to construct a statistical learning model for automated analysis and labeling of biological datasets using pre-trained, unsupervised feature and manifold learning paired with subsequent clustering in order to determine the likely number of biologically meaningful classes. The resultant model will be able to discover relevant features and use the learned dimensionality reduction to identify emerging classes in the data, and in consequence, detect defective or anomalous samples without supervised training or class number knowledge. It can serve as a tool for preliminary analysis of provided samples to identify possible novelty and prevent further misclassification by currently utilized methods. An alternative application is pre-classification of raw datasets for further labelling by experts in the subject matter.

1.3 Research Hypothesis

Based on an extensive literature review in the field of biological pattern recognition, and relying on preliminary work already performed, the following working hypothesis can be proposed:
Automated representation learning paired with subsequent clustering can provide a realistic estimation of the number of biologically significant classes in a collection of biological patterns despite the absence of image content interpretation.

1.4 Assumptions

The research is proposed with the following assumptions:

- The datasets involved in this research are from the biological domain. Scatter patterns and microscopy images are labeled by technicians. It is assumed that overall labeling quality is accurate.
- Third-party open-source software for Python 3.6 programming language paired with Tensorflow 1.6 was used for deep-learning pipelines, Scikit-learn 0.19.1 library was employed for clustering and calculation of assessment parameters. It was assumed that algorithms had been implemented in robust and precise manner.

1.5 Limitations

The research was proposed with the following limitations:

- The machine learning models employed in this study are configured with respect to available computational power. With more powerful graphics processing units (GPU) more convergence of the loss functions may be achieved, which leads to more accurate model.
- The bacteria dataset includes only 10 classes with 100-300 samples per each class. Different combinations of bacteria patterns in a dataset can highly affect clustering quality.
- Hyper-parameters used for the research determined using successful practices available in the literature for similar application areas. The fine-tuning of the parameters will be performed in an empirical manner with respect to time and GPU power constraints.
1.6 Delimitations

The research is proposed with the following delimitations:

- Generalizability of this approach to other biological imaging areas is an open question. On the one hand, the algorithms employed are not hand-crafted anyhow for available datasets. On the other hand, the time and availability constraints will not allow testing of different datasets.

- Since the underlying structure of the test dataset is unknown to trained models according to the hypothesis, the developed tool can be utilized solely to analyze the biological patterns. A classification task without supervised learning is beyond the scope of this work.
CHAPTER 2. REVIEW OF THE LITERATURE

The conducted research requires review of the literature from information technology, computer science, and biomedical engineering. The evolution and application of deep learning is a current emerging phenomenon. State-of-the-art results in computer science are constantly being reported. Rigorous application of the latest techniques requires deep understanding of the processes that power these novel methods. Even during the course of this research some of the methods were updated owing to emerging findings in the deep-learning domain.

2.1 Methodology of the Literature Review Process

The proposed research was based on application of deep-learning techniques to identify the existence of multiple biological patterns in datasets. Most of the papers relevant to the deep-learning domain are dated no earlier than 2012. Despite that, it is a very active area of research with more than 200,000 peer-reviewed articles available in the Purdue Online Library (Purdue University, 2018). There are thousands of publications describing successful application of deep-learning models to biological imaging. However, the proposed research was conducted on two particular datasets from two diverse areas of biological imaging: microscopy in neuroscience and optical imaging for bacteria identification. Hence, to prove the novelty of this approach the list of reviewed literature may be narrowed down to these particular areas. Despite this fact, state-of-the-art technologies should be utilized and their successful application to relative areas should be demonstrated during the literature review process.
2.2 Related Literature Review

The basic knowledge for the proposed research comes from the area of computer science. In one of the most prominent books about deep learning, this approach is described as a solution to "allow computers to learn from experience and understand the world in terms of a hierarchy of concepts, with each concept defined through its relation to simpler concepts" (Goodfellow et al., 2016, p. 2). According to these authors, the term "deep" comes from the idea that a graph representing this structure has many layers. This is a kind of machine learning, which in turn is a common type of artificial intelligence. To my knowledge, the first notable case, when AI successfully challenged a human was in 1997, when IBM's chess-playing machine defeated Garry Kasparov, the world champion at that time (Hsu, 2004). Playing chess, a difficult task for a person, is not in fact challenging for a machine. The game of chess is described by a simple set of determined operations and rules, and successful play is within the machine's computing capability, which allows it to calculate the most optimal game strategies through primitive simulation. A deep-learning approach is intended to tackle intuitive tasks from everyday human life. The difficulty of tasks such as speech or object recognition is that they cannot be formulated as a set of rigid abstract rules. These problems are usually translated to the computer field through a representation learning process. This process is designed to convert raw data, such as images or audio, into a set of descriptive features that are represented as multidimensional vectors (Goodfellow et al., 2016). There is not much innovation in deep learning compared to the traditional machine learning discipline; moreover, there is no consensus on where a machine learning model becomes a deep learning one. Even before the rise of deep learning, most research was conducted on natural images. I presume this happened for two major reasons. First is the variety of potential commercial applications, such as self-driving cars, face recognition in social networks and surveillance systems, and many kinds of digital personal assistants. Second is the fact that the natural images area does
not require expertise in other fields, such as biology, physics, or chemistry. It is a ground point to evaluate and compare different computer science and mathematical models. Moreover, it can easily be compared to human results (Russakovsky et al., 2015).

During the literature review, I encountered the ImageNet dataset in tens of research papers. ImageNet is not only a comprehensive labeled dataset, but an annual contest as well, where researchers from around the world can demonstrate their achievements in natural image recognition. Many state-of-the-art models during recent years demonstrated and proved efficiency evaluating ImageNet (Krizhevsky, Sutskever, & Hinton, 2012; Olga Russakovsky et al., 2015; Szegedy, Ioffe, Vanhoucke, & Alemi, 2016). The project was started at Princeton University and then continued at Stanford University (Deng et al., 2009). To date, it comprises more than 14 million pictures annotated by more than 21,000 different labels (Stanford Vision Lab, 2018).

Prior to 2012, different machine-learning techniques won this competition constantly. In 2011, researchers from Xerox Research Center Europe demonstrated a state-of-the-art result of 25.77% top-5 error rate using a compressed Fisher kernel framework (Perronnin, Liu, Sánchez, & Poirier, 2010). A breakthrough was accomplished in 2012, when Alex Krizhevsky with his students demonstrated a convolutional neural network (CNN) called AlexNet that dropped the error rate of ImageNet classification to 16%, outperforming all competing models by nearly 10% (Krizhevsky et al., 2012). Surprisingly, CNN was described for the first time back in 1989 as a type of neural network with at least one convolutional layer that utilizes mathematical operation – convolution (Goodfellow et al., 2016; Yann LeCun & others, 1989). The raw image is represented as a 2-D grid of pixels that is convolved into an activation map on each layer. One may notice that CNNs were widely adopted only six years ago, despite being first described over 20 years earlier. The supremacy of CNN that emerged in 2012 is owing to the rise in graphical card computational
power (Krizhevsky et al., 2012). In addition, researchers applied an effective combination of convolutional layers with max-pooling layers and used a dropout technique. The dropout technique disables random neurons in the model during training to prevent overfitting. Overfitting occurs when model becomes too complex, losing its approximation characteristics and just fitting all the training data into weighted parameters. Later the efficiency of this technique was demonstrated more formally (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). Another noticeable technical improvement was the use of a rectified linear unit (ReLU) instead of a conventional sigmoid as an activation function in CNN classifiers; the ReLU has been shown to be more efficient in most cases due to sparse representation suitable for naturally sparse data. (Glorot, Bordes, & Bengio, 2011). The activation function is an integral part of any neural network, since it adds a non-linearity to the mathematical model associated with the network.

Furthermore, in 2016 researchers from Google demonstrated a CNN model called Inception-v4 with a 3.08% top-5 error rate in the ImageNet contest (Szegedy et al., 2016). This level of accuracy surpasses reported human-level performance (Russakovsky et al., 2015). All these papers prove the importance of CNN in achieving state-of-the-art performance in image recognition problem.

It is worth noting that Google's Inception-v4 model does not have a principal difference from AlexNet. Improvements in accuracy are achieved through minor changes such as addition of residual connections and more precise estimation of hyper-parameters (Szegedy, Ioffe, Vanhoucke, & Alemi, 2017). Hyper-parameters are an integral part of the building process of the neural network. Their peculiarity is that they are chosen empirically (Luo, 2016).
2.3 Application to Biological Domain

A research group from China proposed a method for application of deep convolutional neural networks to classification and segmentation of histopathology images (Xu et al., 2015). To overcome the issue of scarce datasets, researchers adopted a deep CNN model provided by the Cognitive-Vision team and described in one of the ImageNet contest–related publications (Russakovsky, Deng, Krause, Berg, & Li, 2013). The model was trained using the publicly available labeled dataset ImageNet. Despite the fact the model was trained for natural images, scientists applied it to biological imaging. However, the model was not used end-to-end, but only to extract the features. The feature extraction process can be explained as a transition of raw images into N-dimensional vectors, where N is a hyper-parameter. Using this approach, the Chinese research team outperformed all competitors in the MICCAI 2014 Brain Tumor Digital Pathology Challenge, demonstrating state-of-the-art performance. This work is highly relative to my research, since my intermediate goal is to construct an effective model for feature learning in biological imaging. Similar research took place in 2017 in India (Ajin & Mredhula, 2017). The goal was to identify interstitial lung disease (ILD) using pattern classification applied to computed tomography scans. However, in this case the researchers exploited CNN not for the feature-learning step, but for the classification process. Feature learning was done using linear ternary co-occurrence pattern. Extracted features were classified using various algorithms: artificial neural network (ANN), k-nearest neighbor (KNN), deep CNN, and finally hybrid kernel-based SVM classifier. It is worth noting for clarity that under the ANN term the authors imply classic multilayer perceptron (MLP). Surprisingly, the new SVM-based method produced the most accurate result in classification – 90.52%, outperforming deep CNN by 6.38%. The idea behind the kernel-based SVM classifier is to use SVM with a combination of two kernel functions instead of one: radial basis function and polynomial function. The conclusion that can be drawn from this study is that in some cases deep
CNN can be outperformed by traditional algorithms, when they are precisely hand-crafted for a particular biological problem. However, I lean towards considering that it is rather an exception than a rule.

Wang et al. (2015) conducted a study with an outcome in favor of my conclusion. Their problem was to identify neutrophils, a primary type of immune cell, through image classification. A small dataset of histology tissue slides was used in this research. As in the aforementioned papers, deep CNN was used to learn neutrophils features and Voronoi diagram of clusters to classify needed content. Their findings provided yet additional evidence against hand-crafted features in favor of deep CNN reaching state-of-the-art level of accuracy.

However, another point of view was found in recent research. Coates, Lee and Ng (2011) applied four machine learning algorithms to CIFAR and NORB datasets. The idea of their research was not to compare the performance of the algorithms, but to show the importance of model setup. As a conclusion, they note the significance of selecting the right model parameters. The large number of latent nodes and dense feature vectors was demonstrated to be correlative with model performance. Whitening as a preprocessing step did not yield any noticeable effect in deep-learning models. The important outcome for my research is the superiority of the proposed k-means feature extraction algorithm. In their study, this algorithm outperformed all competitors, demonstrating promising results. As a follow-up to this research, Dundar et al. (2015) revealed surprising results in their paper: a simple and robust k-means feature-learning technique outperformed deep convolutional neural network on an optical imaging dataset obtained through bacteria rapid detection using optical scattering technology (BARDOT). The researchers trained the models using labeled laser-light forward-scatter patterns formed by bacterial colonies. The patterns were represented by grayscale images of 512 by 512 pixels. The dataset included 1833
images with four classes of bacteria. As a result, vector quantization feature learning achieved an overall 97% classification accuracy, outperforming all the other methods (Dundar et al., 2015). My experiment was conducted with the dataset obtained from the same technology, although with a different set of classes and using unsupervised learning or pre-trained features extraction. Based on recent research papers, I do expect a great improvement in accuracy with a deep CNN model compared to the 56% result described in the aforementioned paper.

A research group from Switzerland demonstrated an interesting approach to CNN: they developed a tool to detect mitosis cells in microscopy images (Cireşan, Giusti, Gambardella, & Schmidhuber, 2013). The problem they faced was highlighting the area of mitotic cells in large microscopy images. This task has much in common with the neuroscience microscopy dataset that I am going to use in my research, as well. The researchers converted each image into small patches, in which each pixel of the original image corresponds to a respective patch with this pixel in the center. They trained a CNN model to classify each pixel of the raw image based on its surrounding. One may notice that this approach has much in common with the semantic segmentation problem, which is usually addressed with fully convolutional networks (FNNs) (Long, Shelhamer, & Darrell, 2015). However, owing to the large size of the raw images, FNN cannot be applied to the whole image. The images cannot be split into tiles, since the important attributes may lie on the boundaries. Thus, researchers used the sliding window technique. They participated in the 2012 ICPR Mitosis Detection Contest and won, outperforming competing models by a significant margin (Cireşan et al., 2013). This technique is computationally very expensive; each pixel of the input image requires separate classification of the corresponding patch. In my research, some of the images from the dataset exceed 100 megapixels in size.
All the experiments cited were conducted on biological imaging datasets, although in different subfields. This demonstrates the diversity of the field of biological imaging.

2.4 Autoencoders

Autoencoder is a type of neural network that translates an input datum into a feature vector and then tries to restore the original datum with minimal loss (Bengio, 2009). Owing to its nature, autoencoder is trained in an unsupervised manner; the labeled data are not required, since the loss function expresses how well restored data match the original. One of the major parts of the tool that I plan to develop during my research is a feature-learning model. There are several published papers in which researchers describe successful adoption of an autoencoder to learn features from biological image datasets.

Kallenberg et al. demonstrated state-of-the-art results in breast density segmentation and mammographic risk–scoring problems (2016). Researchers claimed that all prior approaches had been hardly generalizable and required selecting and hand-crafting the features. In contrast, they employed autoencoder to transform patches into feature vectors. Patches were extracted from raw images to train autoencoder in unsupervised manner. As a next step, they trained a simple classifier to output mammographic risk scoring using extracted vectors. The claimed novelty of the research was in formulating "a sparsity regularizer that incorporates both population sparsity and lifetime sparsity" (Kallenberg et al., 2016). It is worth mentioning that simple sparse autoencoders were introduced in 2008 (Boureau & Cun, 2008). Hence, the novelty of this research is in the fact that they constructed a special regularization function to control both population and lifetime sparsity.

Sommer et al. developed a software bundle for detecting novelty in large-scale image datasets (2017). Their approach demonstrated state-of-the-art results in identifying the presence of rare phenotypes without supervised training. In their research paper, they strove to emphasize the
importance of unsupervised learning in the biological imaging domain. They mentioned tens of successful adoptions of CNN models in approaching a variety of medical imaging problems. However, most of them involved labeled datasets that were manually constructed by experts in the subject matter. The core of their novelty-detection framework was a convolutional autoencoder that learns how to effectively convert input images into feature vectors. The feature vector represented the abstract essence of the raw image that serves as an input for a novelty-detection system. The novelty-detection system analyzed some common statistical parameters of feature vectors to distinguish between regular and rare cell phenotypes.

2.5 Common Methods in Related Literature

The overall pipeline for this kind of research is described in detail by Sommer et al. (2017). They exploited deep CNN methods to build an autoencoder and then trained it in an unsupervised fashion. The last step was application to an annotated dataset that had ground truth to assess how well clustering of extracted features can identify novelty in this dataset.

The core of the proposed research is in the representation learning part of the system. All further steps were implemented basically for the purpose of hypothesis testing. During the literature review I encountered different approaches to this feature-learning process. Most of them were implemented using deep learning and convolutional neural networks. Kallenberg et al. exploited a sparse autoencoder approach to the problem of breast density segmentation (2016). Xu et al. adopted a CNN pre-trained on the ImageNet dataset to build a feature-learning pipeline for classification of brain-tumor histopathology images (2015). In a research paper related to neutrophil identification, one of the most relevant pipelines for my experiment was described (Wang et al., 2015). The researchers adopted the current gold-standard of CNN to learn features: a combination of convolutional layers with ReLU activation functions, pooling, and batch
normalization layers. It allowed them to learn features from the dataset for further clustering using a Voronoi diagram of clusters (VDC). During my research I am going to follow the best practices described in one of the most prominent manuscripts concerning deep learning – the book by Goodfellow et al. (2016). The design of the CNN model based on the work described in the paper by Krizhevsky et al. (2012). As a benchmark method for this experiment, I am going to use k-means feature-learning approach described by Coates et al. (2011). This method was later adopted by Dundar et al. and applied to a biological dataset obtained through the same technology as in this research, BARDOT; it outperformed all competing models including deep CNN (2015).

Most of the papers evaluate quality of neural network models on the basis of accuracy percentage: what percent of test data was classified correctly. When it comes to huge datasets like ImageNet, one may use top-5 error accuracy, assuming that the classification is correct if one of the top five outputs is correct (Szegedy et al., 2017). There are two options for setting up to test a dataset. The first is to build two separate datasets, one for training and one for testing. The second, for scarce datasets, is k-fold cross-validation: the whole dataset is split into subsets and the training and testing process is repeated using all combinations of $k-1$ subsets for training and the last one for testing. The results obtained are averaged to get a value for publication (Kohavi, 1995).

2.6 Summary

The wide spread of convolutional neural networks has dramatically changed the area of machine learning associated with image recognition. Researchers around the world apply various modifications of CNN to diverse biological image datasets and report state-of-the-art results. However, the lack of labeled datasets is still an issue that does not have a gold-standard approach. In this research, I am going to evaluate some of the most prominent approaches to unsupervised learning to utilize the most accurate one in further steps of the pipeline.
CHAPTER 3. METHODOLOGY

3.1 Research Framework

The type of research is a computational experiment. The overall pipeline can be divided into two major steps as depicted in Figure 3.1:

1. The representation learning system. The success of the project depends on how well representation learning can capture the intrinsic differences between the analyzed biological patterns. The learned features should be easily clustered by semantic content of the input images; otherwise, they fail to input for the cluster detection module.

2. Clustering to facilitate the discovery of classes in datasets without supervised learning. The synthetic feature vectors are evaluated on their ability to capture the biological information represented in the images and patterns. This process is driven by a separate module employing an unsupervised learning or pre-trained approach. The clustering step can be implemented using a variety of techniques generally divided into two categories: parametric clustering (with hyper-parameter k, which must be evaluated employing an external measure of cluster quality), and non-parametric methods that estimate k automatically.
3.1.1 Feature learning

For the representation learning part of the pipeline, I propose the three most suitable models according to the literature review.

3.1.1.1 Deep convolutional autoencoder

This is a feed-forward unsupervised deep learning model (Goodfellow et al., 2016, p. 4). Raw input images are converted through the series of convolutional, batch normalization, and max-pooling layers into feature vectors. The algorithm of a convolutional layer is described in an early publication by Lecun, Bottou, Bengio, and Haffner (1998). In this research, a receptive field of size 3 by 3 pixels was chosen for all convolutional layers. Two different setups were evaluated with stride 1 and 2. Stride is a hyper-parameter of convolutional layer, which sets the shift of convolution filter during each step of the process.

As an alternative to stride, max-pooling was used to reduce the final size of the feature vector. Max-pooling is a popular subsampling technique in the image recognition domain, which reduces the size of the image using the maximum value of the nearest pixels (Y. LeCun,
Kavukcuoglu, & Farabet, 2010). For this experiment, window of size 2 by 2 pixels was chosen with 2 by 2 pixels stride value to reduce input image size by a factor of four, which is equal to the setting of stride 2 by 2 pixels for convolutional layer. As an alternative to a simple max-pooling layer, max-pooling with argmax function was employed. The argmax function takes multiple input values and outputs the index of maximum. This approach implies simple max-pooling technique; in addition, the position of the maximum pixel is preserved for the subsequent reconstruction in autoencoder. This approach dramatically increase the size of the encoded part of the network, since the array with indices conveys the positions of every fourth pixel of each max-pooling layer. However, it might be helpful to separate encoded features vectors from the index array, since the goal of the research is to learn the underlying patterns from the dataset.

A batch normalization layer is added to the pipeline before the activation function (Ioffe & Szegedy, 2015). The algorithm is represented by a set of equations (3.1). It takes as an input mini-batch $x_i$, which can be the output of any layer of the network, and creates learnable parameters $\gamma, \beta$. It is worth noting, that the last step of this algorithm adds to the network a scale and shift as trainable parameters. In other words, through training any batch normalization layer can be neutralized if needed.
\[ \mu_B \leftarrow \frac{1}{m} \sum_{i=1}^{m} x_i \]
\[ \sigma_B^2 \leftarrow \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_B)^2 \]
\[ \bar{x}_i \leftarrow (x_i - \mu_B) / \sqrt{\sigma_B^2 + \epsilon} \]
\[ y_i \leftarrow \gamma \bar{x}_i + \beta \]

The set of convolutional, batch normalization and max-pooling layers is repeated eight times to reduce the input image of the BARDOT dataset from 512 by 512 pixels to 4000 features.

Equation (3.2) represents the sigmoid function, which was chosen for activation in all layers of the autoencoder.

\[ S(X) = \frac{1}{1 + e^{-x}} \] (3.2)

In the following step, the whole process is repeated in reverse order to reconstruct the dimensions of the original image. On a training step using back propagation, the difference between reconstructed and input image is minimized. The process is depicted on Figure 3.2.
The loss function $l(x)$ for the batch during the training process is represented by the equation (3.3), where $x_{ij}$ is the pixel value of the $j$-th pixel of the $i$-th image of the training mini-batch, $b$ is the size of mini batch, $n$ is the size of image in pixels. The equation is preserved in this form for clarity; that loss function is calculated for the training batch as an average for one image.

$$
l(x) = \frac{\sum_{i=1}^{b} \sum_{j=1}^{n} (D(E(x_{ij}) - x_{ij})^2)}{b} \tag{3.3}
$$

After the model is trained, a test dataset can be converted into feature vectors for future analysis.

3.1.1.2 Pre-trained deep CNN feature extraction

The same set of layers is employed for this neural network. But in this case, as in a conventional CNN classifier, convolutional layers are followed by fully connected layers (see Figure 3.3). For this research, two different setups were evaluated: 600, 200 and 200, 200 neurons for each layer. Equation (3.4) displays the ReLU activation function chosen for this model (Nair & Hinton, 2010).
As a first step, a deep convolutional neural network is trained against ground truth on one dataset and then employed for feature extraction on the test datasets with unseen classes.

The last layer of the conventional deep CNN classifier is cut off after training in order to create a pipeline for converting raw images into feature vectors. The softmax layer, which refers to normalized exponential function, is intended to convert the output of the model into probabilities of each class occurrence and employed in supervised models only. The final feature extraction pipeline is depicted in Figure 3.4. The idea of this feature-learning technique for biological imaging is described in the literature review chapter. There is a difference, though; Wang et al. did not have enough images to train CNN on their own dataset and exploited ImageNet for this purpose (2015). I expect an improvement in result, since supervised pre-training leads to features relevant to specific biological-imaging problems. To put it differently, the model learns the intrinsic features for this particular biological domain filtering all unnecessary details, even though it is not trained at all on target classes.
Figure 3.4. Feature extraction model using trained deep CNN.

To augment the BARDOT dataset and prevent overfitting rotation of input images is added as an option to all CNN-based setups. The structure of BARDOT images is radial, so this option may even improve the overall performance. As a conventional technique to prevent overfitting, a dropout method for latent layers was chosen for neural network models.

3.1.1.3 Representation learning using vector quantization

The method was proposed by Coates et al. and the model was added as a benchmark (2011). I observed the design of the pipeline and its superior results in the same biological domain during the literature review. The pipeline described in the paper is depicted in Figure 3.5. In my research, I cut off the last step, which employs SVM for supervised learning. Description of the feature extraction process is as follows for the training:

1. $m$ random patches are extracted from the training dataset.
2. Those patches are stretched into vectors and preprocessed.
3. K-means clustering algorithm is applied to obtained vectors to produce $k$ centroids.

For the feature extraction step:

1. Input image is converted to $w$-by-$w$ patches using sliding windows with a stride.
2. "Triangle" activation step. The patches are preprocessed, flattened to vectors and classified using $k$ centroids obtained during training.

3. The resulting batch of vectors is pooled to four vectors using the max-pooling method.

4. The resulting four $k$-dimensional vectors are combined to create a $4k$ vector.

Figure 3.5. Pipeline of the vector quantization learning model. Feature learning in the top row; inference in the middle row.

Equation (3.5) represents the activation function mentioned in the second step of the extraction algorithm, where $c^{(k)}$ are learned centroids and, $\mu(z)$ is the mean of the elements of $z$.

$$f_k(x) = \max\{0, \mu(z) - z_k\}$$

$$z_k = \|x - c^{(k)}\|_2$$

As a preprocessing step, standardization by mean and variance is employed. ZCA whitening employed in an original paper does not yield any gain in performance in this biological
dataset (Coates et al., 2011; Krizhevsky, 2009). The same effect regarding to whitening was described in the paper by Dundar et al. (2015).

3.1.2 Subsequent clustering

The goal of the clustering and analysis step is to assess the number of patterns in the testing dataset without supervised learning.

3.1.2.1 Parametric clustering

The parametric clustering method k-means has two goals in this research. The first is validation: the quality of the feature-learning step was compared using ground truth and clustering quality metrics, described in the section on Assessment Instruments. The second is the estimation of the clusters number through the mean silhouette coefficient (Rousseeuw, 1987). It is a conventional method described by Kaufman and Rousseeuw (Kaufman & Rousseeuw, 2009).

3.1.2.2 Non-parametric clustering

This section describes clustering methods, which estimate cluster number through provided data-point analyses. Three popular clustering methods were chosen:

1. K-means clustering with estimation of number of clusters through mean silhouette metric (Pollard & Van Der Laan, 2002).

2. A density-based algorithm for discovering clusters (DBSCAN). The algorithm which determines clusters according to the density of the points (Ester, Kriegel, Sander, & Xu, 2006). Hence, clusters can be of any shape, not just convex.

3. Hierarchical DBSCAN. This is an extended version of DBSCAN that estimates the most stable clustering over epsilon, a hyper-parameter of core-point distance for the DBSCAN (Campello, Moulavi, & Sander, 2013).
3.2 Research Sampling Approach

For this research, I utilized two biological imaging datasets obtained from two ongoing projects in Purdue University.

3.2.1 Scatter patterns collected by bacteria rapid detection using optical scattering technology (BARDOT)

The technology is based on elastic laser-light forward-scatter patterns formed by bacterial colonies impinged by a laser beam of about 1 mm in diameter (Dundar et al., 2015). Individual bacterial colonies are illuminated by the laser at specific wavelength (633nm). This technology outputs forward-scatter signatures, which depend on the colony structure. The results are stored as a gray-scale images of 512 by 512 pixels, one image for each colony. BARDOT implies further classification of collected images to determine the type of bacteria. The particular dataset used in this research consists of 2300 images classified into 10 types of bacteria. The dataset is obtained using simple random sample (SRS) from the database of the Purdue University Cytometry Laboratories collected during last ten years (Robinson, 2017). The provided dataset embraces variations caused by time conditions, light conditions and technicians. The dataset was split into two parts: 1550 images with five classes for training and testing feature-learning models and 750 images and five classes for testing quality of clustering in unseen patterns. This split provides a ground to compare different combinations of feature learning and extraction. Figure 3.6 provides samples for all classes; the names of two of them are replaced by the internal catalog numbers owing to lack of the information.
Figure 3.6. Structure of two BARDOT datasets; (a) dataset A employed for training and testing comprises five classes; (b) dataset B employed for testing comprises five different classes.

3.2.2 Stimulating peripheral activity to relieve conditions (SPARC)

This research is part of ongoing NIH-funded program that aims to advance neuromodulation therapies for accurate neural control of end-organ system function (Jaffey, 2017). The dataset contains 180 microscopy images classified by gastric location and animal. Images are stored in lossless TIFF format and have variable dimensions from 1376 by 1023 up to 27216 by 28212 pixels. One of the SPARC project goals is to analyze patterns of the neurite traces using this dataset. However, the ground-truth labels cannot be obtained at this stage of the project. This aspect limits the validation of the results on the SPARC dataset, owing to the nature of clustering quality assessment methodology.
3.3 Variables

The independent variables are the type of model for representation learning, the clustering algorithm, and the number of classes in the dataset. The dependent variable is the quality of clustering. The intermediate dependent variable is learning loss or accuracy for the representation learning step.

3.4 Assessment Instruments

The quality and validity of the representation learning step was assessed using k-fold cross-validation with $k = 5$ (Kohavi, 1995). The schema is depicted in Figure 3.7. Example data are used to provide numbers for accuracy calculation.

![Figure 3.7. Cross-validation scheme to validate models accuracy.](image)

Based on the literature research, I believe this is the most reliable validation method in machine learning as applied to biological imaging.

3.4.1 Clustering quality

Assessment of the clustering quality is the critical component of this research. The clustering quality metric of the most suitable feature-learning method provides a basis for efficiency evaluation of proposed approach. The feature-learning methods convert raw input
images into multidimensional feature-vector space through learning from the data. Subsequent unsupervised clustering is applied in order to determine to what extent these methods partition an image in a feature-vector space with respect to its semantic content, biological patterns, which is represented by the ground truth labels. Aside from that, a clustering algorithm per se may affect the aforementioned metrics if the resulting clusters have uncertain borders.

To be a source of conclusions, the clustering quality metric should not depend on cluster number, classes number, their ratio, or the number of samples. Based on my literature review, I decided to employ V-measure, homogeneity and completeness (Rosenberg & Hirschberg, 2007). These are the key metrics for clustering quality assessment with respect to ground truth data implemented in the most popular machine learning library for the Python programming language scikit-learn (Pedregosa et al., 2011).

3.4.1.1 V-measure, homogeneity, completeness

V-measure was introduced by Rosenberg and Hirschberg (2007). After the clustering step of the pipeline, given N data points and two sets which partition these points and represent ground-truth n classes and m clusters respectively. Notations are displayed in equation (3.6).

\[ C = \{c_i | i = 1, \ldots, n\}, K = \{k_j | j = 1, \ldots, m\} \]  

(3.6)

Let \( A = \{a_{ij}\} \) be the set of numbers, where \( a_{ij} \) represents the count of data points that belongs to \( c_i \) class and \( k_j \) cluster. The homogeneity, an entropy-based measure, is reflected in the equations (3.7) and (3.8).

\[
h = \begin{cases} 
1, & H(C, K) = 0 \\
1 - \frac{H(C | K)}{H(C)}, & \text{else}
\end{cases}
\]  

(3.7)

where
\[ H(C|K) = -\sum_{k=1}^{|K|} \sum_{c=1}^{|C|} \frac{a_{ck}}{N} \log \frac{a_{ck}}{\sum_{c=1}^{|C|} a_{ck}} \]

\[ H(C) = -\sum_{c=1}^{|C|} \frac{\sum_{k=1}^{|K|} a_{ck}}{n} \log \frac{\sum_{k=1}^{|K|} a_{ck}}{n} \]

(3.8)

In the worst case, homogeneity is equals to 0.0, hence perfectly homogeneous clustering corresponds to 1.0. The intuition behind this metric is that it reflects how well different classes can be distinguished using clustering.

Completeness represented in equations (3.9) and (3.10), is complementary to homogeneity. It reflects the distribution of classes among clusters. In the perfect case, when completeness equals to one, all points from the class belongs to one cluster, with one cluster for each class. Even if there is only one cluster that comprises the whole dataset, it is still perfectly complete. In contrast, if each cluster includes points from each class with distribution equals to the distribution of clusters, completeness is 0.0.

\[ c = \begin{cases} 1, & H(K, C) = 0 \\ 1 - \frac{H(K|C)}{H(K)}, & \text{else} \end{cases} \]

(3.9)

where

\[ H(K|C) = -\sum_{c=1}^{|C|} \sum_{k=1}^{|K|} \frac{a_{ck}}{N} \log \frac{a_{ck}}{\sum_{c=1}^{|C|} a_{ck}} \]

\[ H(K) = -\sum_{k=1}^{|K|} \frac{\sum_{c=1}^{|C|} a_{ck}}{n} \log \frac{\sum_{c=1}^{|C|} a_{ck}}{n} \]

(3.10)

Finally, V-measure can be represented by equation (3.11) as a weighted harmonic mean of homogeneity and completeness.
\[ V = \frac{(1 + \beta) \cdot h \cdot c}{\beta \cdot h + c} \] (3.11)

In various use-case scenarios different values of \( \beta \) are demanded. For simplicity and generalizability of the results the value was set to one, implying that homogeneity and completeness are equally important.

3.4.1.2 Mean silhouette coefficient

Mean silhouette coefficient (MSC) is a parameter that reflects the compactness and separation of clusters without reliance on ground truth, assessing how well each point matches its current cluster compared to if it were moved to the closest one (Pollard & Van Der Laan, 2002). The silhouette number for point \( i \) can be represented by the following equation (3.12).

\[
s(i) = \begin{cases} 
1 - \frac{a(i)}{b(i)}, & a(i) < b(i) \\
0, & a(i) = b(i) \\
\frac{b(i)}{a(i)} - 1, & a(i) > b(i)
\end{cases}
\] (3.12)

where \( a(i) \) and \( b(i) \) are the mean distances from a point to elements of the same cluster and to elements of the closest cluster respectively (Rousseeuw, 1987). MSC is a value in the range \([-1; 1]\). Values greater than zero correspond to partitioning, where on average all clusters are separated. However, when values are close to zero, nearest clusters are very close to each other and clustering may not be robust.

3.4.1.3 Graphical aid for interpretation of clustering

For the purpose of visualization of the results in two- and three-dimensional space two major dimension reduction techniques were employed:
1. T-distributed stochastic neighbor embedding (t-SNE). The intuition behind this method is to represent the clusters in a limited dimension space, preserving the ratio of distances between clusters (Maaten & Hinton, 2008).

2. Principal component analysis (PCA). Singular value decomposition of the data is employed to reduce dimensions (Tipping & Bishop, 2007).

3.4.2 SPARC dataset methodology

Owing to the lack of labeled data for the SPARC dataset the applied methodology has some deviations from the BARDOT. The proposed tool can be applied on a per-image basis. One of the images was processed manually to provide neurite trace as a ground truth. The image was split into 503 by 959 patches of 8 by 8 pixels with a 2 by 2 pixels stride. The resulting images were arranged into a labeled dataset with 482,377 patches. The labeling is binary, since the neurites patterns are the only areas of interest in this dataset. Owing to this preference, the main clustering should be followed by post-processing to merge all the clusters except the neurites one. This can be achieved using representative shape descriptors applied to the resulting clustering. For this particular case, solidity demonstrates appropriate results. The cluster with the lowest solidity represented by equation (3.13) has a great fit with the neurite map (Russ, 2007, p. 582).

\[
Solidity = \frac{Area}{Convex Area}
\]  

(3.13)

Owing to the size of the dataset and the fact the neurite map in the provided dataset usually occupies 5-10% of the image, the main clustering quality metrics are not applicable. For this reason, a conventional confusion matrix was employed (Stehman, 1997). The efficiency can be evaluated by assessing the percentage of identified neurite map, while taking into consideration false-positive identification. In other words, precision and recall metrics for the neurite class.
CHAPTER 4. PRESENTATION OF DATA & FINDINGS

The previous chapter introduced the computational pipelines that were employed in this research. In this section the actual setups for both datasets and data finding will be described.

4.1 BARDOT Setup and Findings

As described, for the BARDOT dataset three major feature extraction techniques were employed: convolutional autoencoder, pre-trained CNN and a representation learning using vector quantization. -

4.1.1 Convolutional autoencoder

4.1.1.1 Setup configuration

Table 4.1 presents the configuration for convolutional layers used for all experiments conducted on the BARDOT dataset. Constant 3 by3 patch size is used throughout all convolutional layers. The activation map of each next layer is subsampled by the factor of four either by convolutional stride or by max-pooling. The eventual encoded vector size is 2 by 2 by 1000 features, which is further flattened on the extraction step.
Table 4.1. Convolutional layer configuration of autoencoder for BARDOT dataset

<table>
<thead>
<tr>
<th>Number</th>
<th>Receptive field</th>
<th>Input size</th>
<th>Feature layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3 x 3</td>
<td>512 x 512</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>3 x 3</td>
<td>256 x 256</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>3 x 3</td>
<td>128 x 128</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>3 x 3</td>
<td>64 x 64</td>
<td>80</td>
</tr>
<tr>
<td>5</td>
<td>3 x 3</td>
<td>32 x 32</td>
<td>160</td>
</tr>
<tr>
<td>6</td>
<td>3 x 3</td>
<td>16 x 16</td>
<td>320</td>
</tr>
<tr>
<td>7</td>
<td>3 x 3</td>
<td>8 x 8</td>
<td>500</td>
</tr>
<tr>
<td>8</td>
<td>3 x 3</td>
<td>4 x 4</td>
<td>1000</td>
</tr>
</tbody>
</table>

Table 4.2 presents four different hyper-parameter setups that were tested on this dataset. The first goal was to establish the most effective subsampling technique for the biological domain, since it does not have much in common with natural images. The second one was to determine if the image rotation improves performance, since BARDOT images have radial structure, and in prior works researchers occasionally employed an unfolding technique.

Table 4.2. Hyper-parameter setups of autoencoder for BARDOT dataset.

<table>
<thead>
<tr>
<th>Setup</th>
<th>Subsampling</th>
<th>Image rotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>max-pooling</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>max-pooling with argmax</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>convolutional stride</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>max-pooling with argmax</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Each configuration was trained on 1240 images of dataset A and then evaluated on 310 testing images from dataset A and 750 testing images from dataset B. Results are obtained through K-fold cross-validation with K=5.
4.1.1.2 Presentation of the results

Figure 4.1 demonstrates the process of loss function convergence during training of setups described in Table 4.2. Outliers at the start of the training are excluded from the chart scaling. Employment of max-pooling layers with argmax function clearly demonstrates better recovery capabilities of the autoencoder compared to simple max-pooling or convolutional stride. By definition, max-pooling layers with argmax preservation greatly expand the size of the encoded feature vectors by recovering the positions of pooled neurons. Image rotation does not demonstrate any improvement in image recovery capability of the autoencoder.

![Loss function curves against training step.](image)

*Figure 4.1. Loss function curves against training step.*

For this amount of training data improvement in minimization of loss function stops after 12,000 training steps with 50 images in each mini-batch. The "flat" curve is a signal of model convergence. Visually it is almost impossible to distinguish between images reconstructed by autoencoder with restoration using argmax function, however simple max-pooling layers demonstrate a quite blurred result (see Figure 4.2).
Figure 4.2. Restoration capabilities on BARDOT dataset of AE with simple max-pooling: (a) input, (b) output. AE with max-pooling and argmax: (c) input, (d) output.

To assess the efficiency of employed models, clustering quality against ground truth data should be compared using a fixed clustering method – k-means with cluster numbers equal to the ground truth. Table 4.3 demonstrates results for autoencoders using cross-validation on BARDOT dataset A.

Table 4.3. Cross-validated clustering quality metrics for autoencoder on BARDOT dataset A.

<table>
<thead>
<tr>
<th>AE Setup</th>
<th>Homogeneity</th>
<th>Completeness</th>
<th>V-measure</th>
<th>Silhouette</th>
</tr>
</thead>
<tbody>
<tr>
<td>max-pooling</td>
<td>0.9756</td>
<td>0.9762</td>
<td>0.9759</td>
<td>0.3055</td>
</tr>
<tr>
<td>MP with argmax</td>
<td>0.8068</td>
<td>0.8335</td>
<td>0.8197</td>
<td>0.4996</td>
</tr>
<tr>
<td>convolutional stride</td>
<td>0.8746</td>
<td>0.9169</td>
<td>0.8949</td>
<td>0.2672</td>
</tr>
<tr>
<td>MP, argmax, image rotation</td>
<td>0.8355</td>
<td>0.8482</td>
<td>0.8417</td>
<td>0.4605</td>
</tr>
</tbody>
</table>

Results from BARDOT dataset A imply that the autoencoder model has been trained on the same dataset split into training and testing parts. Hence, the model was trained to restore known type of bacteria patterns. As seen from the results, configuration with simple max-pooling subsampling layers demonstrates the best results out of all autoencoder setups, although the loss function after convergence had the worst results. It turned out that the model with the worst result
in image reconstruction from feature vector problem provided supreme results in clustering quality. Figure 4.3 demonstrates labeled feature vectors plotted in 2-D space using t-SNE dimensionality reduction for this setup.

![Figure 4.3: Scatter plot of labeled feature vectors obtained from BARDOT dataset A with t-SNE dimensionality reduction.](image)

As the next step, performance of these autoencoders setups was evaluated on BARDOT dataset B. This dataset consists of five unseen for models classes. Table 4.4 demonstrates the results.

<table>
<thead>
<tr>
<th>AE Setup</th>
<th>Homogeneity</th>
<th>Completeness</th>
<th>V-measure</th>
<th>Silhouette</th>
</tr>
</thead>
<tbody>
<tr>
<td>max-pooling</td>
<td>0.9551</td>
<td>0.9555</td>
<td>0.9553</td>
<td>0.4341</td>
</tr>
<tr>
<td>MP with argmax</td>
<td>0.7540</td>
<td>0.7726</td>
<td>0.7632</td>
<td>0.4756</td>
</tr>
<tr>
<td>convolutional stride</td>
<td>0.9269</td>
<td>0.9285</td>
<td>0.9277</td>
<td>0.3709</td>
</tr>
<tr>
<td>MP, argmax, image rotation</td>
<td>0.7355</td>
<td>0.7652</td>
<td>0.7500</td>
<td>0.4418</td>
</tr>
</tbody>
</table>
Autoencoder setup with simple max-pooling subsampling demonstrates supremacy on the dataset even with unseen patterns. Figure 4.4 helps visually assess the difference between t-SNE embedding for ground truth labeling and k-means clustering. T-SNE embedding demonstrates a great result in separation of data points with respect to ground truth labels, the only mismatch is where a set of points from class #5 is combined with class #2. Moreover, class #4 might have two distinguishable patterns in it. K-means clustering confuses small numbers of data points from most of the classes with class #2 or cluster #5.

*Figure 4.4. Autoencoder results on BARDOT dataset B: (a) ground truth with t-SNE embedding, (b) k-means clustering with K=5 and t-SNE embedding.*

Using silhouette score to establish number of clusters using k-means, DBSCAN and HDBSCAN algorithms with cross-validation, we can estimate the number of clusters in both BARDOT datasets given autoencoder-produced features (see Table 4.5). DBSCAN clustering demonstrates the most accurate result. It provides a stable output for the number of clusters in BARDOT dataset A. Paired with a high V-measure score, accurate clustering with this non-parametric method can be expected. Figure 4.5 confirms this expectation. Even though two clusters have a fair number of outliers, the overall clustering matches ground-truth classes.
Table 4.5. Cluster-number estimation based on extracted features using autoencoder.

<table>
<thead>
<tr>
<th>Clustering type</th>
<th>Dataset A</th>
<th>Dataset B</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means (Silhouette score)</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>DBSCAN</td>
<td>5</td>
<td>5.4</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>3.6</td>
<td>6</td>
</tr>
</tbody>
</table>

Figure 4.5. T-SNE embedding of feature-vectors obtained on BARDOT dataset A with convolutional autoencoder; (a) ground truth for five classes; (b) DBSCAN clustering with five clusters. Red dashes demonstrate outliers for cluster #4 and #5.

4.1.2 Pre-trained deep CNN feature extraction

4.1.2.1 Setup configuration

This method cannot be evaluated on the dataset with the same classes that it was trained on, according to the statement of the problem of this research. However, it can possibly serve as a tool to discover unknown biological patterns, and thus compete with other methods when applied to datasets with unexposed bacteria classes. The deep CNN model consists of six convolutional and two fully-connected layers. Table 4.6 demonstrates the convolutional layers setup, since it is
constant across all configurations. Two select appropriate hyper-parameters four most promising setups were chosen (see Table 4.7).

Table 4.6. Convolutional layers configuration of pre-trained CNN for BARDOT dataset.

<table>
<thead>
<tr>
<th>Number</th>
<th>Receptive field</th>
<th>Input size</th>
<th>Feature layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3 x 3</td>
<td>512 x 512</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>3 x 3</td>
<td>256 x 256</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>3 x 3</td>
<td>128 x 128</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>3 x 3</td>
<td>64 x 64</td>
<td>70</td>
</tr>
<tr>
<td>5</td>
<td>3 x 3</td>
<td>32 x 32</td>
<td>120</td>
</tr>
<tr>
<td>6</td>
<td>3 x 3</td>
<td>16 x 16</td>
<td>200</td>
</tr>
</tbody>
</table>

Table 4.7. Hyper-parameter setups of pre-trained CNN for BARDOT dataset.

<table>
<thead>
<tr>
<th>Setup</th>
<th>Config</th>
<th>Image rotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>max-pooling, 200+200 FC</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>conv stride, 200+200 FC</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>conv stride, 200+200 FC</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>conv stride, 600+200 FC</td>
<td>No</td>
</tr>
</tbody>
</table>

As with autoencoder, the model was trained on dataset A. However, cross-validated testing results on dataset A may only suite to demonstrate the nature of the features extracted using this type of model. Dataset B was evaluated as a whole, following the main methodology. Model training took 20,000 loss-function-minimization steps with 0.01 as a training step.

4.1.2.2 Presentation of the results

Figure 4.6 presents loss function and accuracy graphs during the training process for setup #3, since graphs from the other three setups look the same. All of the setups achieved 100% cross-validated accuracy in classification of dataset A.
Although the setup in Table 4.8 cannot be compared with other models, since the model was trained on dataset A in a supervised manner, it can provide a useful insight, when comparing with the results obtained on unseen dataset B.

*Table 4.8.* Cross-validated clustering quality metrics achieved on BARDOT dataset A using features extracted with CNN classifier.

<table>
<thead>
<tr>
<th>AE Setup</th>
<th>V-measure</th>
<th>Silhouette</th>
</tr>
</thead>
<tbody>
<tr>
<td>max-pooling, 200+200 FC</td>
<td>1.0</td>
<td>0.4538</td>
</tr>
<tr>
<td>conv stride, 200+200 FC</td>
<td>1.0</td>
<td>0.8038</td>
</tr>
<tr>
<td>conv stride, 200+200 FC, image rotation</td>
<td>1.0</td>
<td>0.8033</td>
</tr>
<tr>
<td>conv stride, 600+200 FC</td>
<td>1.0</td>
<td>0.8478</td>
</tr>
</tbody>
</table>

Table 4.8 demonstrates that the tightest and the most separated clusters are achieved with convolutional stride as subsampling technique and an extended fully connected layer. All setups achieve perfect V-measure scores, an expected result, since the accuracy of all classifiers is 100% after cross-validation.
Table 4.9 provides clustering quality metrics obtained on dataset B. The model with convolutional stride as a subsampling technique and an extended fully-connected layer demonstrated the best results in both V-measure and silhouette.

Table 4.9. Cross-validated clustering quality metrics achieved on BARDOT dataset B using features extracted with pre-trained CNN.

<table>
<thead>
<tr>
<th>AE Setup</th>
<th>Homogeneity</th>
<th>Completeness</th>
<th>V-measure</th>
<th>Silhouette</th>
</tr>
</thead>
<tbody>
<tr>
<td>max-pooling, 200+200 FC</td>
<td>0.7316</td>
<td>0.7688</td>
<td>0.7494</td>
<td>0.4538</td>
</tr>
<tr>
<td>conv stride, 200+200 FC</td>
<td>0.7718</td>
<td>0.7853</td>
<td>0.7784</td>
<td>0.5492</td>
</tr>
<tr>
<td>conv stride, 200+200 FC, image rotation</td>
<td>0.6956</td>
<td>0.7341</td>
<td>0.7139</td>
<td>0.5264</td>
</tr>
<tr>
<td>conv stride, 600+200 FC</td>
<td>0.8052</td>
<td>0.8128</td>
<td>0.8090</td>
<td>0.5663</td>
</tr>
</tbody>
</table>

When it comes to estimating number of clusters, this model provides robust features for dataset A (see Table 4.10). It was trained with ground-truth data to distinguish between these patterns. However, for unseen patterns in dataset B only HDBSCAN can estimate a result that is close to truth. K-means and DBSCAN tend to identify more classes than expected. Although such results may be a sign of the existence of sub patterns in each bacteria class, the homogeneity score is lower than for the autoencoder model. Hence, this model is less accurate in estimating biological patterns in the BARDOT dataset.

Table 4.10. Cross-validated cluster-number estimation based on extracted features with pre-trained CNN.

<table>
<thead>
<tr>
<th>Clustering type</th>
<th>Dataset A</th>
<th>Dataset B</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means (Silhouette score)</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>DBSCAN</td>
<td>5.8</td>
<td>11.8</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>5.0</td>
<td>5.8</td>
</tr>
</tbody>
</table>
4.1.3 Representation learning using vector quantization

Table 4.11 represents the hyper parameters chosen for the feature learning model according to Coates et al. with corrections by Dundar et al. (2011; 2015).

Table 4.11. Hyper-parameters for representation learning using vector quantization.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means centroids</td>
<td>1000</td>
</tr>
<tr>
<td>Training steps</td>
<td>5000</td>
</tr>
<tr>
<td>Receptive field</td>
<td>6 by 6</td>
</tr>
</tbody>
</table>

Table 4.12 demonstrates clustering quality metrics. As in previous cases, the model was trained on classes from dataset A and then evaluated on both datasets. Estimation of the number of clusters in both BARDOT datasets demonstrates results in favor of HDBSCAN (see Table 4.13).

Table 4.12. Cross-validated clustering quality metrics achieved on BARDOT datasets using features extracted with representation learning using vector quantization.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Homogeneity</th>
<th>Completeness</th>
<th>V-measure</th>
<th>Silhouette</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.8302</td>
<td>0.8433</td>
<td>0.8367</td>
<td>0.5750</td>
</tr>
<tr>
<td>B</td>
<td>0.9687</td>
<td>0.9689</td>
<td>0.9687</td>
<td>0.5724</td>
</tr>
</tbody>
</table>

Table 4.13. Cross-validated cluster-number estimation based on extracted features with representation learning using vector quantization.

<table>
<thead>
<tr>
<th>Clustering type</th>
<th>Dataset A</th>
<th>Dataset B</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means (Silhouette score)</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>DBSCAN</td>
<td>6.2</td>
<td>4</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>5</td>
<td>5.4</td>
</tr>
</tbody>
</table>

HDBSCAN successfully estimated the number of clusters for both datasets. The method output five clusters for all cross-validation runs for dataset A. Such a stable outcome can be
perceived as result of tight and separated dataset points in a feature-vector space. However, comparison to ground-truth classes reveals serious misclassification (see Figure 4.7).

![Figure 4.7](image)

*Figure 4.7.* T-SNE embedding of feature vectors obtained through vector quantization on BARDOT dataset A. (a) ground truth for five classes; (b) HDBSCAN clustering with five clusters; red circles highlight mismatch in clustering.

4.2 **Setup and Findings for SPARC Dataset**

To evaluate the generalization of the proposed approach, the feature-learning method that had demonstrated the most accurate and robust results was employed. Table 4.14 demonstrates configuration of convolutional layers for the autoencoder applied to SPARC dataset.

<table>
<thead>
<tr>
<th>Number</th>
<th>Receptive field</th>
<th>Input size</th>
<th>Feature layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3 x 3</td>
<td>8 x 8</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>3 x 3</td>
<td>4 x 4</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>3 x 3</td>
<td>2 x 2</td>
<td>40</td>
</tr>
</tbody>
</table>

For the clustering step, only k-means was employed, since the problem requires clustering for the whole dataset, which is unachievable by non-parametric methods during in this research.
A convolutional autoencoder was trained through 15,000 SGD steps. Figure 4.8 presents a confusion matrix for binary clustering given the neurite map. The minimum solidity method successfully identified the cluster for the neurite map.

<table>
<thead>
<tr>
<th>Predicted labels</th>
<th>Background</th>
<th>Neurite trace</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Background</strong></td>
<td>433268</td>
<td>12857</td>
</tr>
<tr>
<td><strong>Neurite trace</strong></td>
<td>5115</td>
<td>31137</td>
</tr>
</tbody>
</table>

*Figure 4.8.* Confusion matrix obtained from clustering SPARC dataset on a per-image basis. Results are validated using Monte Carlo cross-validation.

According to the confusion matrix the applied approach was able to identify 85.89% of the neurite trace, having 29.22% of pixels identified as false-positive. The values are validated using Monte Carlo cross-validation (Dubitzky, Granzow, & Berrar, 2007, p. 178). Considering Figure 4.9, the binary clustering results can be assessed visually. The fact that the confusion matrix provides 29.22% of mismatch does not conflict with the overall goal of the applied approach, which is identification of patterns, not semantic segmentation.
(a) original image from SPARC dataset with neurite highlighted with Biotin

(b) highlighted clusters obtained using k-means

*Figure 4.9.* Application of autoencoder with k-means clustering (silhouette-based estimation of $K$ parameter) on SPARC dataset.
CHAPTER 5. CONCLUSIONS AND DISCUSSION

This paper proposed application of different feature-learning techniques with subsequent clustering as a tool to estimate and analyze the existence of biological patterns. The objective of this research was to evaluate the performance of chosen feature-learning methods through a series of computational experiments on datasets from two different biological domains. Consolidated results are presented in Table 5.1.

Table 5.1. Consolidated, cross-validated results of clusters quality for all three proposed methods for BARDOT datasets.

<table>
<thead>
<tr>
<th>Feature learning</th>
<th>V-measure Dataset A</th>
<th>V-measure Dataset B</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN Autoencoder</td>
<td>0.9759</td>
<td>0.9553</td>
</tr>
<tr>
<td>CNN Classifier</td>
<td>-</td>
<td>0.8090</td>
</tr>
<tr>
<td>Vector quantization</td>
<td>0.8367</td>
<td>0.9687</td>
</tr>
</tbody>
</table>

Figure 5.1 demonstrates four boxplots comparing cross-validation distributions for vector quantization and convolutional autoencoder and both datasets. The variance of the distribution involving dataset B is much smaller owing to employment of the whole dataset during each run. Results of significance test for both datasets provide an evidence that for these datasets vector quantization and convolutional autoencoder have significantly different clustering quality results. The difference of cross-validated metrics for BARDOT dataset B is 0.0134 in favor of vector quantization; the same value for BARDOT dataset A is 0.1392 in favor of convolutional autoencoder. Despite unsupervised training on the BARDOT dataset A, the feature learning using vector quantization demonstrated a significant drop in accuracy comparing to unexposed dataset. This result might be due to visually similar patterns in different classes.
Figure 5.1. Boxplots for distributions over cross-validation runs for (1) vector quantization on dataset A, (2) autoencoder on dataset A, (3) vector quantization on dataset B, (4) autoencoder on dataset B.

Based on the results, it can be concluded that a convolutional autoencoder provides a combination of accuracy and robustness and can facilitate unsupervised identification of unknown patterns in biological datasets. It may be utilized as a feature extraction instrument for similar purposes in further research or construction of analytical tools. The recommended type of subsampling of input image through the convolutional layers is max-pooling without preservation of original positions. The computational graph for the model is provided in Appendix A. However, simple method of feature learning using vector quantization also demonstrated high precision on bacteria datasets. It is worth mentioning that even though the training of this algorithm is much faster than for deep-learning models, the inference is computationally harder. For example, the feature extraction method employed leads to a set of operations with roughly 260 million of four-byte values for inference on one 512-by-512 grayscale image.
During the literature review, the most promising method for feature extraction was identified as a deep CNN classifier trained on one set of bacteria classes to employ it for further feature extraction on different patterns. However, the readouts provide evidence that plain supervised training produces a feature extraction pipeline with features extremely bound to known bacteria patterns. During training, the loss function of the pure classification problem was minimized almost perfectly. It may lead to unstable vectors positioning in feature space when extracting features of unknown classes – visually different images can be tied together (see Figure 5.2).

![Figure 5.2. Zoomed region of PCA 3-D embedding of features extracted from BARDOT dataset B, using CNN classifier trained on BARDOT dataset A with a different set of classes. Colors of images represent ground truth classes. Red arrows point to visually different bacteria that were positioned close to *Salmonella anatum* (violet-colored) class. PCA embedding describes 95.6% of variance in 200-D vector space.](image)

Even though clustering methods that estimate number of clusters through learning from data provided quite accurate results, this approach may still be applied only for analysis and estimation purposes. Given zero information regarding patterns arrangement between biological classes, it makes no sense to expect from any clustering algorithm the robust and accurate
separation of data points with respect to those classes. However, this technique may be employed to observe different patterns inside one class: learning of intrinsic features of the biological classes, their relative positioning, detection of outliers and deviations that might be the signal of a foodborne illness outbreak when it comes to the bacteria identification problem or simply a defective sample which may cause further issues as well. Moreover, to my knowledge, current classification methods employed in BARDOT technology do not include a deep CNN model. Taking into consideration 100% accuracy demonstrated in this research, the adoption of a convolutional network for bacteria classification is recommended.

The application of this method to a completely different microscopy dataset provides a good ground in favor of generalization of this approach to other biological-imaging domains. However, it requires completely different preprocessing and measurement. As a future application to image data obtained during SPARC project, this method can provide a feature extraction tool to classify different neurite patterns. The identified neurite cluster described in Chapter 4, can be easily converted into feature vector in different ways. For instance, representative shape descriptors can be employed for this problem: form-factor, elongation, convexity, solidity, etc. (Russ, 2007, p. 582). As a next step, original images of 10-100 megapixels can be classified into different types with respect to highlighted neurite trace and a little amount of labeled data.

In future work, new emerging deep-learning models and methods can be applied, since this is a current active area of research. Following the intuition behind the autoencoder and deep-learning classifier, the better results might be obtained through merging these two concepts together using one unite loss function and conventional SGD for minimization.
APPENDIX A. COMPUTATIONAL GRAPH FOR CONVOLUTIONAL AUTOENCODER
REFERENCES


