HOKKAIDO UNIVERSITY

Title	A study on data-efficient learning and its medical applications
Author(s)	李，広
Degree Grantor	北海道大学
Degree Name	博士(情報科学)
Dissertation Number	甲第15666号
Issue Date	2023-09-25
DOI	https://doi.org/10.14943/doctoral.k15666
Doc URL	https://hdl.handle.net/2115/90861
Type	doctoral thesis
File Information	Li_Guang.pdf

Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

A thesis for the degree of Doctor of Philosophy

A Study on Data-Efficient Learning and Its Medical Applications

データエフィシェントラーニングとその医療応用に関する研究

Guang Li

Graduate School of Information Science and Technology

Hokkaido University

September, 2023

Contents

1 Introduction 8

1.1 Background . . . 8

1.2 Proposition in this Thesis . . . 9

1.3 Organization of this Thesis . . . 10

2 Related Works 13

2.1 Introduction . . . 13

2.2 Dataset Complexity Assessment Methods . . . 13

2.3 Coreset Selection and Dataset Distillation Methods . . . 16

2.4 Self-supervised Learning Methods . . . 17

2.5 Problems to Be Solved in this Thesis . . . 19

2.6 Conclusion . . . 20

3 Dataset Complexity Assessment Based on Spectral Clustering 21

3.1 Introduction . . . 21

3.2 Method . . . 21

3.2.1 Dimension reduction . . . 23

3.2.2 Similarity matrix construction . . . 23

3.2.3 Spectral clustering . . . 24

3.2.4 Dataset complexity calculation . . . 26

3.3 Experiments . . . 27

3.3.1 Datasets . . . 27

3.3.2 Comparison with benchmark and the state-of-the-art methods . . . 28

3.3.3 The effectiveness of pretrained DCNN feature extractors . . . 31

3.3.4 Interclass distance visualization . . . 33

3.3.5 The influence of the image feature's dimension . . . 35

3.4 Conclusion . . . 36

4 Compressed Gastric Data Generation Based on Soft-Label Dataset Distillation 37

4.1 Introduction . . . 37

4.2 Method . . . 38

4.2.1 Training data preprocessing 38

4.2.2 Compressed gastric image generation 39

4.2.3 Full gastric image classification 44

4.3 Experiments 45

4.3.1 Experimental settings 45

4.3.2 Demonstration of the effectiveness of dataset reduction 46

4.3.3 Demonstration of the effectiveness of model compression 48

4.3.4 Minimum number of compressed images 50

4.4 Conclusion 52

5 Self-Supervised Transfer Learning for Automatic COVID-19 Detection 53

5.1 Introduction 53

5.2 Method 54

5.3 Experiments 56

5.3.1 Dataset and settings 56

5.3.2 Experimental results 59

5.4 Conclusion 60

6 High-Accuracy Automatic COVID-19 Detection via Self-Supervised Learning and Batch Knowledge Ensembling 61

6.1 Introduction 61

6.2 Method 62

6.2.1 Phase I: Self-Supervised Learning-based Pretraining 62

6.2.2 Phase II: Batch Knowledge Ensembling-based Fine-tuning 65

6.3 Experiments 67

6.3.1 Dataset and Settings 67

6.3.2 Test Accuracy on the Large COVID-19 CXR Dataset 71

6.3.3 Test Accuracy on the COVID5K Dataset 73

6.3.4 Exploring the Impact of Hyperparameters on Experimental Results . . . 73

6.3.5 Performance Comparison with Existing Methods 75

6.4 Conclusion 76

7 Efficient gastritis detection based on self-supervised representation learning 77

7.1 Introduction 77

7.2 Method 78

7.2.1 Gastric X-ray image preprocessing 78

7.2.2 Self-supervised learning 78

7.2.3 Fine-tuning and gastritis detection 82
7.3 Experiments 82
7.3.1 Dataset 83
7.3.2 Implementation 83
7.3.3 Results 86
7.4 Conclusion 86

8 Conclusions 89
8.1 Overview of the Proposition in this Thesis 89
8.2 Future Directions 90

Acknowledgements 92

Bibliography 93

Achievements of the Author 103

List of Figures

2.1 Research map of related researches. 20

3.1 Overview of the proposed method. 22

3.2 Concept of the proposed method. 27

3.3 Laplacian spectrum of the six 10-class datasets (Comb.). 30

3.4 Laplacian spectrum of the six 10-class datasets (EfficientNet-B4 and t-SNE). . . 32

3.5 Interclass distance of different datasets: (a) mnist, (b) cifar10, and (c) stl10. . . . 34

4.1 Overview of the proposed method. 38

4.2 Compressed image generated in SLDD (3). 47

4.3 Compressed image generated in DD (3). 48

4.4 Compressed image generated in SLDD (1). 48

5.1 Overview of the proposed method. 54

5.2 Test results of COVID-19 detection in different data volumes: (a) HM and (b) Acc. 57

5.3 Confusion matrix for the best model of our method. 58

5.4 Grad-CAM++ visual explanations of the proposed method. 58

6.1 Overview of the proposed method. 63

6.2 Examples of CXR images in the large COVID-19 CXR dataset [1]: (a) COVID-19, (b) Lung Opacity (c) Normal, and (d) Viral Pneumonia. 68

6.3 Examples of CXR images in the COVID5K dataset [2]: (a) COVID-19 , (b) Normal. 68

6.4 Best performance confusion matrix of our method. (a): ResNet50, (b): ResNet18. 69

6.5 Test accuracy in different annotated data volumes: (a) HM of ResNet50, (b) HM of ResNet18, (c) Accuracy of ResNet50, and (d) Accuracy of ResNet18. 71

6.6 Best performance confusion matrix of our method. (a): ResNet50, (b): ResNet18. 72

7.1 Overview of the proposed method. 79

7.2 Details of the partitioned datasets used in the present study. SSL denotes self-supervised learning process. 84

List of Tables

3.1 Pearson correlation and p-value between the complexity and the test error rates of the six 10-class datasets. 29

3.2 The network structure of the 9-layer CNN autoencoder. 30

3.3 Test error rates for three DCNN models on the six 10-class datasets [3]. 31

3.4 Complexity of the six 10-class datasets (Comb.). 31

3.5 Pearson correlation and p-value between the complexity and the test error rates of the six 10-class datasets. 32

3.6 Pearson correlation between the complexity and the test error rates of the five 10-class datasets (excluding one of the six datasets). 33

3.7 Pearson correlation and p-value between the complexity and the test error rates of the six 10-class datasets with different reduced dimensions. 36

4.1 Comparison of dataset distillation (ResNet18) and ResNet18. 47

4.2 Comparison of different models with batch normalization (bn) and without batch normalization (no_bn). 49

4.3 Memory footprints of different models. Memory denotes saving all of the parameters of a model. Memory$*$ denotes saving batch normalization parameters and distilled results. 49

4.4 Comparison of the minimum number of compressed images of different models. . 50

4.5 Parameters of different models. Parameter denotes the number of model parameters. Image denotes the minimum number of compressed images. 51

5.1 Details of the COVID-19 chest X-ray dataset [9] used in our study. “C”: COVID-19, “L”: Lung Opacity,“N”: Normal, and “V”: Viral Pneumonia. 56

5.2 Test results of COVID-19 detection. 56

6.1 Details of the large COVID-19 CXR dataset [1]. 67

6.2 Details of the COVID5K dataset [2]. 69

6.3 Test accuracy on the large COVID-19 CXR Dataset. 70

6.4 Test accuracy in different annotated data volumes when compared with vision transformer-based methods. 70

6.5 Test accuracy on the COVID5K dataset. 72

6.6 Evaluation results on the changes of the ensembling weight ω and the batch size N. 75

6.7 Evaluation results on the changes of the temperature τ and the weighting factor λ. 75

6.8 Performance comparison with the existing methods. 76

7.1 Hyperparameters of the proposed method. 84

7.2 Comparison with the state-of-the-art self-supervised learning methods. 87

7.3 Comparison with our previous methods. 88

Chapter 1

Introduction

The background and purpose of this thesis and the organization of this thesis are presented in this chapter.

1.1 Background

Deep learning [4, 5] has experienced remarkable advancements and demonstrated remarkable achievements in various domains, such as computer vision [6], natural language processing [7], and speech recognition [8]. In recent years, prominent deep learning models, including AlexNet [9], ResNet [10], BERT [11], Wave2vec [12], ViT [13], CLIP [14], Stable Diffusion [15], and ChatGPT [16], have been developed and relied upon large-scale datasets for training. However, working with such large datasets poses significant challenges in terms of storage, transmission, and preprocessing [17]. Additionally, training on large-scale datasets requires extensive computational resources, often involving thousands of GPU hours to achieve good performance [18]. To address these challenges, this thesis focuses on investigating data-efficient learning methods.

Data-efficient learning is a subfield of machine learning that focuses on training models with limited amounts of data while maintaining high performance [19]. Traditional machine learning algorithms often require large datasets to generalize well and make accurate predictions. However, in many real-world scenarios, collecting and labeling massive amounts of data can be time-consuming, expensive, or even impractical [20]. Data-efficient learning aims to overcome these limitations and develop methods that can effectively learn from small or scarce datasets.

One method of data-efficient learning is transfer learning [21], where a pre-trained model on a large dataset is fine-tuned on a smaller target dataset. By leveraging the knowledge learned from the larger dataset, the model can quickly adapt to the new task with fewer training examples. This method has been successfully applied in various domains, including computer vision, natural language processing, and speech recognition. Another method of data-efficient learning is active learning [22], which involves selecting informative samples from a large pool of unlabeled data and actively querying human experts to label those samples. The labeled samples are then used to train a model, and the process iterates, gradually improving the model's performance with a minimal amount of labeled data. Active learning can significantly reduce the labeling effort and achieve good performance with a small labeled dataset. Furthermore, techniques such as semi-supervised learning [23] and weakly supervised learning [24] also contribute to data-efficient learning. In semi-supervised learning, models are trained using a combination of labeled and unlabeled data, where the unlabeled data provides additional information to improve the model's generalization. Weakly supervised learning, on the other hand, deals with tasks where only partial or noisy supervision is available, allowing models to learn from imperfect labels or weak annotations.

Data-efficient learning is a rapidly evolving field driven by the necessity to address real-world problems with limited data availability. While existing methods can alleviate some of the challenges posed by large-scale datasets, they inherently possess limitations when applied to certain scenarios. For instance, constructing new datasets requires careful consideration of their complexity to effectively train neural networks. Moreover, existing methods may not be suitable for situations involving extremely limited data or labels. Therefore, there is a need for exploring stronger data-efficient learning methods to address these limitations.

1.2 Proposition in this Thesis

The purpose of this thesis is to construct new datasets more efficiently and to enhance the learning capabilities of models when facing extremely limited data or labels. To achieve this goal, the thesis proposes a novel data-efficient learning method consisting of the following three stages. The first stage involves assessing the complexity of datasets by analyzing their characteristics and properties. Understanding the complexities of a dataset allows researchers to make well-informed choices regarding model architecture, training strategies, and data augmentation techniques that are appropriate for that particular dataset. This stage plays a crucial role in optimizing the learning process and achieving superior performance with limited data. Building upon the dataset complexity assessment, the second stage introduces the concept of dataset distillation. Dataset distillation leverages knowledge from a larger, labeled dataset to distill it into a smaller, more compact dataset. The distilled dataset retains the most relevant information that is essential for the target task. This stage can enhance data processing efficiency and avoid overfitting or noise from the large dataset. Lastly, the third stage explores self-supervised learning as a data-efficient learning method. Self-supervised learning involves training models to solve pretext tasks using unlabeled data, with labels generated automatically or through heuristics. The learned representations from these pretext tasks can then be transferred to the target task, effectively utilizing the large amounts of unlabeled data to improve performance. This stage can reduce reliance on labeled data while still achieving competitive results. With the incorporation of the three stages, the newly proposed data-efficient learning method can effectively address the existing challenges.

The contributions of this thesis can be summarized as follows:

- The thesis introduces a new data-efficient learning method that encompasses three stages: dataset complexity assessment, dataset distillation, and self-supervised learning. The proposed method aims to construct new datasets with improved efficiency and enhance model learning capabilities, particularly when dealing with severely limited data or labels.

- The effectiveness of the proposed method is evaluated on both natural image datasets and medical image datasets. The proposed method extends the existing knowledge and techniques in data-efficient learning and provides valuable insights for researchers and practitioners working in this area.

1.3 Organization of this Thesis

The remainder of this thesis is organized as follows.

In Chapter 2, related works of data-efficient learning are presented and problems to be solved are clarified.

In Chapter 3, a dataset complexity assessment method based on spectral clustering is presented. The training process of deep convolutional neural networks is iterative and time-consuming because of hyperparameter uncertainty and the domain shift introduced by different datasets, especially for complex medical datasets. Hence, it is meaningful to predict classification performance by assessing the complexity of datasets effectively before training DCNN models. The proposed method can evaluate the dataset's complexity effectively before training DCNN models.

In Chapter 4, a method of generation of compressed gastric images based on soft-label dataset distillation for efficient anonymous medical data sharing is presented. Sharing of medical data is needed to enable the cross-agency flow of healthcare information and the construction of high-accuracy computer-aided diagnosis systems. The proposed method not only compresses a whole medical dataset into only one compressed soft-label patch image but also reduces the size of a trained model to a few hundredths of its original size, which can improve the efficiency of medical data sharing. The compressed images obtained after distillation have been completely anonymized and therefore do not contain private information of the patients, which can improve the security of medical data sharing. Furthermore, the proposed method can achieve high classification performance with only a small number of compressed images.

In Chapter 5, a self-supervised transfer learning method for COVID-19 detection from chest X-ray images is presented. Under the global pandemic Coronavirus Disease 2019, computer-aided diagnosis for COVID-19 fast detection and patient triage is becoming critical. The proposed method can learn discriminative representations from chest X-ray images by combining transfer learning and self-supervised learning. The method can achieve promising results on the largest open COVID-19 chest X-ray dataset.

In Chapter 6, for boosting COVID-19 detection accuracy, a novel method based on self-supervised learning and self-knowledge distillation is presented. This method is an extended version of the method proposed in Chapter 6. The proposed method can use self-knowledge of images based on similarities of their visual features. The proposed method can achieve promising results on the largest open COVID-19 chest X-ray dataset and another unbalanced COVID-19 chest X-ray dataset.

In Chapter 7, a self-supervised learning method for learning discriminative representations from gastric X-ray images is presented. Manually annotating gastric X-ray images for gastritis detection is time-consuming and expensive because it typically requires expert knowledge. The proposed method is based on a teacher–student architecture and cross-view and cross-model losses, which can perform explicit self-supervised learning and learn discriminative representations from gastric X-ray images. The proposed method can achieve a high patient-level gastritis detection performance with only a few annotations.

In Chapter 8, the conclusions of this thesis and the future directions are discussed.

The methods proposed in each chapter of the thesis correspond to the authors' research achievements summarized at the end. In Chapter 3, the proposed method, [A-1], is an extension of the method presented in [B-2]. Chapter 4 introduces the method proposed in [A-2], which builds upon the method described in [B-1]. The method presented in Chapter 5, [A-3], is an extension of the method discussed in [B-5]. In Chapter 6, the authors propose the method [A-4], which extends the method outlined in [B-6]. Lastly, Chapter 7 introduces the method [A-5], which is an extension of the method described in [B-7].

Chapter 2

Related Works

2.1 Introduction

This chapter shows the research related to this thesis. Subsection 2.2 describes the related works on dataset complexity assessment, subsection 2.3 shows the previous works on coreset selection and dataset distillation, and subsection 2.4 shows the previous works on dataset distillation. Next, subsection 2.5 clarifies the problems to be solved in this thesis. Finally, subsection 2.6 concludes this chapter.

2.2 Dataset Complexity Assessment Methods

Reference [25]

This study introduces a novel approach to assess dataset complexity by proposing twelve descriptors: F1, F2, F3, N1, N2, N3, N4, L1, L2, L3, T1, and T2. These descriptors serve different purposes in evaluating the complexity of a dataset. The first three descriptors, F1, F2, and F3, focus on feature-based methods. F1 represents the maximum Fisher's discriminant ratio, quantifying the discriminative power of features in separating classes. F2 measures the interclass overlap of feature distributions, providing insights into the separability of classes. F3 identifies the most efficient feature in separating classes by finding the maximum value. The descriptors N1, N2, N3, and N4 are neighborhood methods that examine the presence and density of classes within local neighborhoods. These descriptors offer information about the arrangement and distribution of classes, contributing to the overall complexity assessment. On the other hand, L1, L2, and L3 are linear methods that evaluate the potential linear separability of classes. These descriptors quantify whether classes can be separated effectively through linear decision boundaries. Lastly, T1 and T2 represent topological methods. T1 measures the total number of hyperspheres that can be fitted into the feature space of a class, providing insights into the topological complexity. T2, on the other hand, divides the number of examples in the dataset by their dimension, offering an indicator of the sparsity and dimensionality of the dataset.

Reference [26]

This paper introduces a novel distance measure for images, which can also be seen as a complexity assessment method, termed IMage Euclidean Distance (IMED), which takes into consideration the spatial relationships of pixels. Unlike the traditional Euclidean distance, IMED exhibits robustness to small perturbations in images. The proposed IMED distance is further applied to image recognition tasks. To evaluate the effectiveness of the IMED distance measure, experiments are conducted using the Face Recognition Technology database and two state-of-the-art face identification algorithms. The results demonstrate consistent performance improvements when the algorithms are embedded with the new metric compared to their original versions.

Reference [27]

This research draws inspiration from the analysis of ill-posed regression problems by Elden and the interpretation of linear discriminant analysis as a mean square error classifier. By employing Singular Value Decomposition analysis, this study introduces a discriminatory power spectrum as a means of assessing data complexity in undersampled classification problems. The discriminatory power spectrum quantifies the concentration of discriminatory power within the dataset. Through experimentation with five real-life biomedical datasets of increasing difficulty, the research demonstrates the relationship between data complexity and the performance of regularized linear classifiers.

Reference [28]

The complexity measures implemented in this study are derived from the descriptions provided by [25]. While the initial definitions of these measures have been revised and updated, some modifications have been made to adapt them to the specific context of this research. Originally, these measures were designed for two-class datasets and primarily applied to problems with continuous attributes. Nominal or categorical attributes were numerically encoded and treated as continuous, as most complexity measures rely on distance functions between attributes. This study has extended the majority of these measures to handle multi-class datasets. This extension allows for a broader application of the complexity measures to datasets with multiple classes. Additionally, this study implemented the most relevant distance functions for both continuous and nominal attributes.

Reference [29]

This paper focuses on investigating the influence of noise on the complexity of classification problems. The study aims to analyze the sensitivity of various complexity indices in the presence of different levels of label noise. Geometric, statistical, and structural measures derived from the data are employed to characterize the complexity of a classification dataset. By examining the behavior of these measures when noise is introduced into the dataset, the researchers gain insights into the impact of noise on data complexity. The experimental results demonstrate that certain complexity measures exhibit higher sensitivity to the addition of noise compared to others. These findings highlight the potential use of these sensitive measures in developing preprocessing techniques for noise identification and designing novel algorithms that are tolerant to label noise. Additionally, the study presents preliminary results on a new noise identification filter that leverages two complexity measures that demonstrated higher sensitivity to the presence of label noise.

Reference [3]

The proposed methodology in this study focuses on characterizing the overlap in feature distribution among different classes in an image dataset. It specifically calculates the complexity of the dataset by analyzing the eigenvalues of a Laplacian matrix, which is derived from the similarity matrix representing the relationships between the classes. The size of the Laplacian spectrum is utilized as a measure of dataset complexity, where a larger spectrum indicates a higher degree of overlap between classes. The method achieved SOTA results in several datasets.

2.3 Coreset Selection and Dataset Distillation Methods

Reference [30]

In this paper, efficient algorithms for approximating the k-means and k-medians problems in Euclidean metrics have been proposed to solve the coreset problem. To achieve a high-quality approximation, it is crucial to compute a coreset that is as small as possible while still capturing the essential characteristics of the input. In low-dimensional scenarios, coresets enable the development of approximation algorithms with a running time that is linear or nearly linear, with an additional term depending only on the size of the coreset.

Reference [31]

This paper introduces a novel method to significantly reduce the size of a large set of data points in a high-dimensional Euclidean space Rd to a small set of weighted points while preserving the accuracy of various data analysis tasks performed on the reduced set. This reduced set, commonly known as a coreset, provides approximate solutions for the original set in tasks such as computing principal components or performing k-means clustering. The proposed method is based on projecting the points onto a low-dimensional subspace and reducing the cardinality of the projected points using established techniques. This approach can be applied to various data analysis techniques, including k-means clustering, principal component analysis, and subspace clustering.

Reference [32]

This paper introduces a novel training strategy called iCaRL to tackle the challenge of catastrophic forgetting. Unlike previous approaches that were limited to fixed data representations and incompatible with deep learning architectures, iCaRL enables class-incremental learning. It achieves this by allowing the presence of training data for only a small number of classes at any given time, while also providing the flexibility to progressively incorporate new classes into the learning process.

Reference [33]

To tackle the catastrophic forgetting issue, this paper proposes an incremental learning approach for deep neural networks. The proposed method leverages new data while utilizing only a small exemplar set consisting of samples from the old classes. The method accomplishes this by employing a loss function composed of two components: a distillation measure to retain the knowledge acquired from the old classes, and a cross-entropy loss to learn the new classes. Importantly, this approach enables end-to-end incremental training, meaning the data representation and classifier are learned jointly.

Reference [34]

This paper proposed a framework called generalization-based data subset selection (Glister) for efficient and robust learning. Glister addresses the challenge of efficient and robust training by formulating it as a mixed discrete-continuous bi-level optimization problem. The objective is to select a subset of the training data that maximizes the log likelihood on a held-out validation set. Glister introduces an iterative online algorithm. This algorithm performs data selection iteratively while updating the model parameters, making it applicable to any loss-based learning algorithm.

Reference [35]

This paper first introduced the concept of dataset distillation and presents an algorithm that employs backpropagation through optimization steps to accomplish dataset distillation. Dataset distillation involves the synthesis of a compact dataset that enables models trained on it to achieve high performance on a larger original dataset. The goal of a dataset distillation algorithm is to take a large real dataset as input and generate a small synthetic distilled dataset. The effectiveness of the distilled dataset is evaluated by testing models trained on it using a separate real dataset. A high-quality distilled dataset has diverse applications, such as continual learning, privacy preservation, and neural architecture search.

2.4 Self-supervised Learning Methods

Reference [36]

This paper explores the problem of image representation learning in a self-supervised manner, without the need for human annotation. The approach utilizes the concept of self-supervision by training a convolutional neural network (CNN) to solve Jigsaw puzzles as a pretext task. This pretext task does not require manual labeling, making it an effective approach for learning representations. The trained CNN can then be repurposed for object classification and detection tasks. To ensure compatibility across tasks, the paper introduces a context-free network (CFN), which is a siamese CNN. The CFN takes image tiles as input and employs a mechanism that limits the receptive field or context of its early processing units to one tile at a time. Remarkably, the CFN achieves comparable semantic learning capabilities to AlexNet while utilizing fewer parameters. By training the CFN on Jigsaw puzzles, the network learns a feature mapping of object parts and their spatial arrangement.

Reference [37]

This paper introduces a novel approach to learning image features using CNN trained to identify the 2D rotation applied to input images. Despite its simplicity, this task proves to be highly effective in guiding semantic feature learning, as evidenced by comprehensive qualitative and quantitative analyses. The proposed method is thoroughly evaluated on diverse unsupervised feature learning benchmarks, consistently outperforming existing techniques and achieving state-of-the-art results.

Reference [38]

The aim of self-supervised learning from images is to create meaningful image representations without relying on semantic annotations. While many existing pretext tasks in self-supervised learning produce representations that are covariant with image transformations, This paper thinks that semantic representations should be invariant under such transformations. To tackle this challenge, this paper introduces Pretext-Invariant Representation Learning (PIRL), a method that learns invariant representations through pretext tasks. Specifically, when applying PIRL to a popular pretext task involving solving jigsaw puzzles, experimental results demonstrate a significant improvement in the semantic quality of the learned image representations through PIRL.

Reference [39]

This paper introduces SimCLR, a straightforward framework for contrastive learning of visual representations. Unlike existing methods, SimCLR avoids the need for specialized architectures or a memory bank. To gain insights into the effective learning of representations through contrastive prediction tasks. The findings highlight the critical role of data augmentation composition, the benefits of introducing a learnable nonlinear transformation between the representation and contrastive loss, and the advantages of larger batch sizes and more training steps in contrastive learning compared to supervised learning. By leveraging these insights, SimCLR achieves significant improvements over previous methods for self-supervised and semi-supervised learning on ImageNet.

Reference [40]

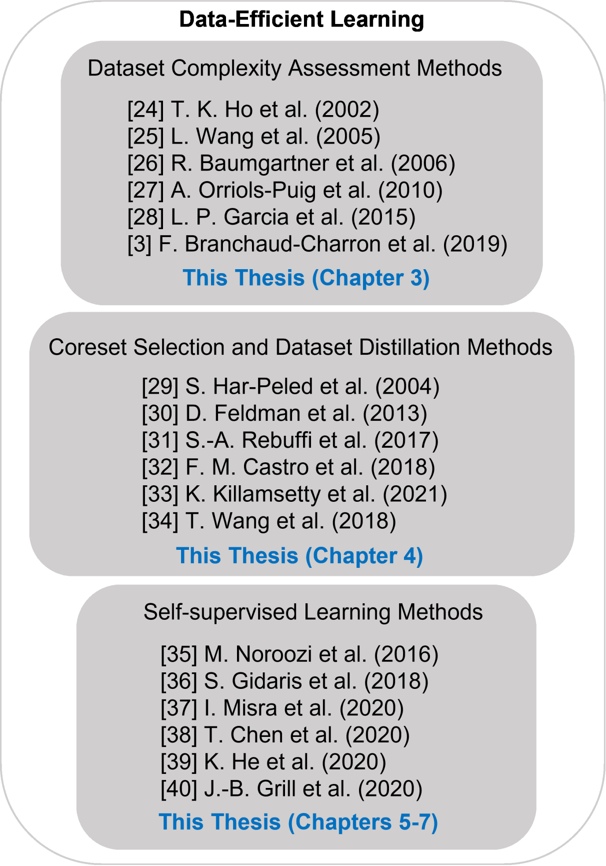
This paper introduces Momentum Contrast (MoCo) as an approach to unsupervised visual representation learning. Viewing contrastive learning as a form of dictionary look-up, MoCo constructs a dynamic dictionary using a queue and a moving-averaged encoder. This allows the creation of a large and consistent dictionary on-the-fly, which greatly facilitates contrastive unsupervised learning. MoCo achieves competitive results on the widely-used linear protocol for ImageNet classification. Moreover, the learned representations in MoCo exhibit strong transferability to downstream tasks.

Reference [41]

This paper presents Bootstrap Your Own Latent (BYOL), a novel approach to self-supervised image representation learning. BYOL utilizes two neural networks: the online network and the target network, which interact and learn from each other. Given an augmented view of an image, the online network is trained to predict the target network representation of the same image under a different augmented view. Concurrently, BYOL updates the target network using a slow-moving average of the online network. Notably, BYOL achieves a new state-of-the-art performance without relying on negative pairs, which are commonly used in existing methods.

2.5 Problems to Be Solved in this Thesis

Based on the related research outlined, the problems to be solved in this thesis will be clarified. The purpose of this study is to enhance the learning capabilities of models with limited data and improve their performance. To achieve this goal, this thesis proposed corresponding methods in dataset complexity assessment, dataset distillation, and self-supervised learning. The effectiveness of these proposed methods is evaluated in the medical domain, where data availability is



Data-Efficient Learning

Dataset Complexity Assessment Methods

[24] T. K. Ho et al. (2002)
[25] L. Wang et al. (2005)
[26] R. Baumgartner et al. (2006)
[27] A. Orriols-Puig et al. (2010)
[28] L. P. Garcia et al. (2015)
[3] F. Branchaud-Charron et al. (2019)
This Thesis (Chapter 3)

Coreset Selection and Dataset Distillation Methods

[29] S. Har-Peled et al. (2004)
[30] D. Feldman et al. (2013)
[31] S.-A. Rebuffi et al. (2017)
[32] F. M. Castro et al. (2018)
[33] K. Killamsetty et al. (2021)
[34] T. Wang et al. (2018)
This Thesis (Chapter 4)

Self-supervised Learning Methods

[35] M. Noroozi et al. (2016)
[36] S. Gidaris et al. (2018)
[37] I. Misra et al. (2020)
[38] T. Chen et al. (2020)
[39] K. He et al. (2020)
[40] J.-B. Grill et al. (2020)
This Thesis (Chapters 5-7)

Figure 2.1: Research map of related researches.

often limited due to privacy concerns and the scarcity of expert annotations. By applying these data-efficient learning methods to medical datasets, the thesis aims to demonstrate their efficacy in improving the performance of models in real-world applications. Figure 2.1 shows a research map of related research that summarizes the above.

2.6 Conclusion

This chapter explained the research related to this thesis. Furthermore, the problems to be solved in this thesis are clarified.

Chapter 3

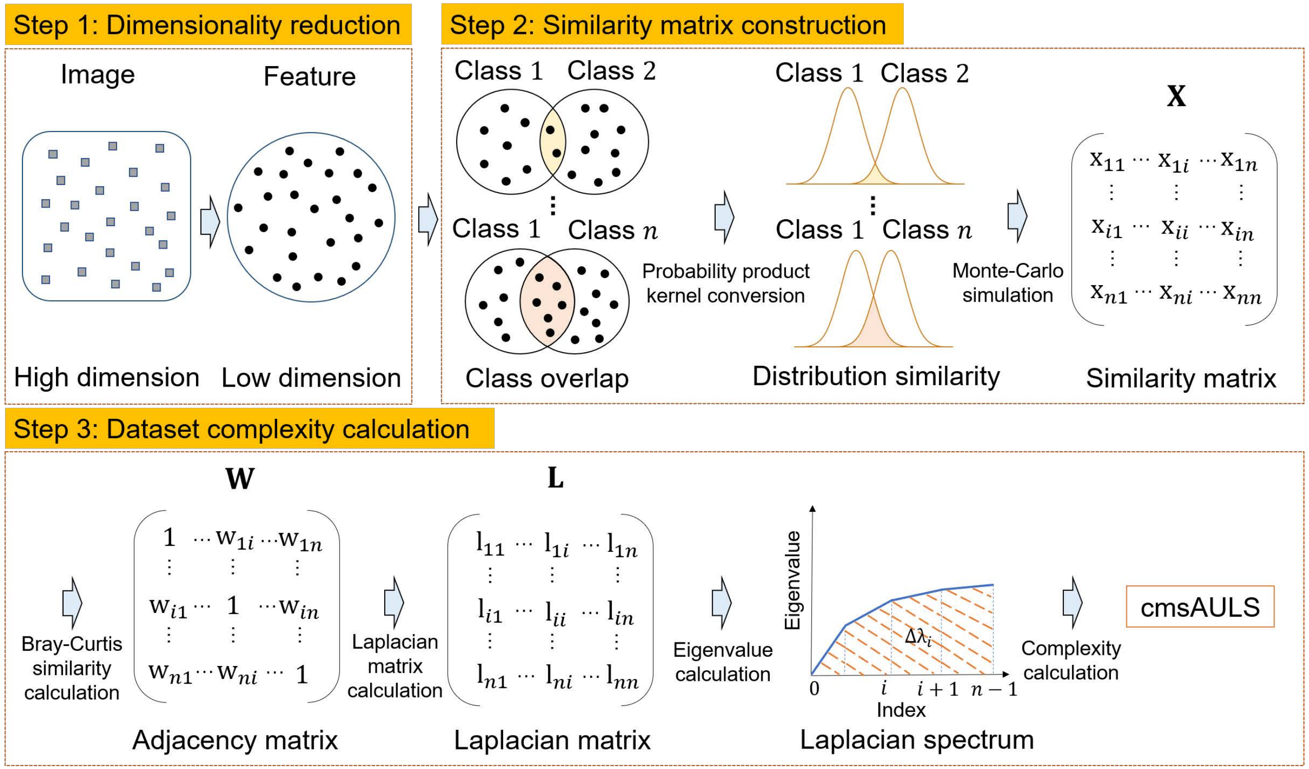
Dataset Complexity Assessment Based on Spectral Clustering

3.1 Introduction

Dataset complexity assessment aims to predict the performance of classification models on a given dataset by calculating its complexity. This assessment not only helps in selecting appropriate classifiers but also aids in dataset reduction. Training deep convolutional neural networks (DCNNs) involves an iterative and time-consuming process due to hyperparameter uncertainty and domain shift caused by different datasets. Therefore, it is crucial to effectively assess dataset complexity before training DCNN models to predict classification performance accurately. In this chapter, we propose a novel method called cumulative maximum scaled Area Under Laplacian Spectrum (cmsAULS). This method demonstrates state-of-the-art performance in assessing dataset complexity across six different datasets. By employing cmsAULS, we can achieve reliable predictions of classification performance, thus enabling efficient model training and selection.

3.2 Method

In subsection 3.2.1, we provide a comprehensive explanation of the dimension reduction phase. This step aims to reduce the dimensionality of the dataset while preserving its essential characteristics and minimizing information loss. Moving forward, subsection 3.2.2 illustrates the process of constructing a similarity matrix that captures the relationships between classes within the



Step 1: Dimensionality reduction
Image
Feature
High dimension
Low dimension
Step 2: Similarity matrix construction
Class 1
Class 2
Class 1
Class n
Class overlap
Probability product kernel conversion
Class 1
Class 2
Class 1
Class n
Distribution similarity
Monte-Carlo simulation
X
$x_{11} \cdots x_{1i} \cdots x_{1n}$
$x_{i1} \cdots x_{ii} \cdots x_{in}$
$x_{n1} \cdots x_{ni} \cdots x_{nn}$
Similarity matrix
Step 3: Dataset complexity calculation
Bray-Curtis similarity calculation
W
$1 \cdots w_{1i} \cdots w_{1n}$
$w_{i1} \cdots 1 \cdots w_{in}$
$w_{n1} \cdots w_{ni} \cdots 1$
Adjacency matrix
Laplacian matrix calculation
L
$l_{11} \cdots l_{1i} \cdots l_{1n}$
$l_{i1} \cdots l_{ii} \cdots l_{in}$
$l_{n1} \cdots l_{ni} \cdots l_{nn}$
Laplacian matrix
Eigenvalue calculation
Eigenvalue
$\Delta\lambda_i$
0
i
$i+1$
$n-1$
Index
Laplacian spectrum
Complexity calculation
cmsAULS

Figure 3.1: Overview of the proposed method.

dataset. This matrix serves as the basis for evaluating the complexity of the dataset. Furthermore, in subsection 3.2.3, we delve into the relationship between spectral clustering and dataset complexity. Spectral clustering is utilized to identify underlying structures within the dataset, contributing to the assessment of its complexity. Finally, subsection 3.2.4 outlines the methodology for calculating the dataset complexity. This step involves incorporating the dimension reduction results, similarity matrix, and spectral clustering information to obtain a comprehensive measure of the dataset's complexity.

3.2.1 Dimension reduction

To handle high-dimensional image data, it is necessary to transform them into a lower-dimensional space while preserving their inherent characteristics. Let us consider an input data point x, and its embedding is defined as $\psi(x) \in \mathbb{R}^d$, where d represents the dimension of the downscaled feature space. The function ψ can encompass various dimension reduction methods, such as autoencoder [42], t-SNE [43], or PCA [44]. These methods enable the transformation of the input data x into a lower-dimensional representation.

3.2.2 Similarity matrix construction

The degree of overlap between classes serves as an indicator of the complexity of an image dataset for classification tasks, as highlighted in prior work [3]. Accordingly, the proposed method determines the dataset complexity by assessing the overlap between classes. Although the dataset may contain multiple classes (n), our approach focuses on analyzing the overlap between any two classes. By considering a pair of classes $\mathcal{A}$ and $\mathcal{B}$, the goal is to compute the overlap across the entire dataset. Drawing from the integral measure of the Gaussian mixture model [45], the overlap between classes $\mathcal{A}$ and $\mathcal{B}$ refers to the collective region in the image feature space where the conditional probability $P(\psi(x_t) \mid \mathcal{B})$ exceeds $P(\psi(x_t) \mid \mathcal{A})$ for any $\psi(x_t)$ belonging to class $\mathcal{A}$. Based on this understanding, we can define the class overlap as follows:

$$\int_{\mathbb{R}^d} \min(P(\psi(x) \mid \mathcal{A}), P(\psi(x) \mid \mathcal{B}))\, d\psi(x), \tag{3.1}$$

where distributions $P(\psi(x) \mid \mathcal{A})$ and $P(\psi(x) \mid \mathcal{B})$ represent the probability distributions of the image feature $\psi(x)$ belonging to classes $\mathcal{A}$ and $\mathcal{B}$, respectively. Directly calculating the integral in Eq. (1) can be highly complex and computationally intensive. However, leveraging the strong correlation between class overlap and the similarity of data distributions, we can employ the probability product kernel [46] as a surrogate for Eq. (1). The surrogate expression is as follows:

$$\int_{\mathbb{R}^d} P(\psi(x) \mid \mathcal{A})^\rho \, P(\psi(x) \mid \mathcal{B})^\rho \, d\psi(x). \tag{3.2}$$

When the parameter $\rho = 1$, the inner product between the two distributions corresponds to the expectation of one distribution under the other. In other words, it represents either $\mathbb{E}_{P(\psi(x)|\mathcal{A})}[P(\psi(x) \mid \mathcal{B})]$ or $\mathbb{E}_{P(\psi(x)|\mathcal{B})}[P(\psi(x) \mid \mathcal{A})]$. However, directly calculating the expectation becomes inefficient when dealing with datasets containing a large number of images for classes $\mathcal{A}$ and $\mathcal{B}$. To address this challenge, we employ the Monte Carlo method [47] to approximate the expectation calculation. This method allows us to estimate the expectation by sampling from the respective distributions. The approximation process is as follows:

$$\mathbb{E}_{P(\psi(x)|\mathcal{A})}[P(\psi(x) \mid \mathcal{B})] \approx \frac{1}{M} \sum_{m=1}^{M} p(\psi(x_m) \mid \mathcal{B}). \tag{3.3}$$

In the proposed method, we select M samples $\psi(x_m)$ $(m = 1, 2, \cdots, M)$ randomly from class $\mathcal{A}$, and $p(\psi(x_m) \mid \mathcal{B})$ represents the probability of $\psi(x_m)$ belonging to class $\mathcal{B}$. By calculating the expectation between all classes, we construct the similarity matrix $\mathbf{X} \in \mathbb{R}^{n \times n}$, where n is the number of classes in the dataset. Furthermore, we utilize a k-nearest estimator to approximate $p(\psi(x_m) \mid \mathcal{B})$ as follows:

$$p(\psi(x_m) \mid \mathcal{B}) = \frac{K}{EV}, \tag{3.4}$$

where the parameter K represents the number of neighbors of $\psi(x_m)$ within class $\mathcal{B}$. Furthermore, we define E as the number of samples randomly selected from class $\mathcal{B}$, and V represents the volume of the hypercube that contains the k closest neighbors around $\psi(x_m)$ within class $\mathcal{B}$.

3.2.3 Spectral clustering

In this section, we explore the relationship between spectral clustering and dataset complexity. We utilize the calculated similarity matrix $\mathbf{X}$, which contains comprehensive information about the complexity of the entire dataset. To extract meaningful insights from $\mathbf{X}$, we employ spectral clustering theory [48]. We consider an undirected similarity graph G, comprising nodes and edges. The weight ($w_{ij} \geq 0$) of an edge connecting nodes i and j represents their proximity or similarity. These edge weights are stored in an $n \times n$ adjacency matrix $\mathbf{W}$, where n denotes the total number of nodes. The objective of spectral clustering is to partition G into a collection of subgraphs $\{G_1, \cdots, G_i, \cdots, G_j, \cdots, G_r\}$ such that the weights of edges between different subgraphs are minimized. Mathematically, we aim to find a partition that satisfies $G_i \cap G_j = \varnothing$ for all $i \neq j$, and $G_1 \cup \cdots, \cup, G_r = G$. To achieve this optimal partition, it is crucial to minimize the cost of the cut between subgraphs, denoted as $\mathrm{Cut}(G_1, \cdots, G_r) = \sum w_{ij}$ for i and j, which means the cost is calculated as the sum of weights w_{ij} for all edges connecting nodes in different subgraphs.

Spectral clustering offers a solution to the partition problem by leveraging the Laplacian spectrum. To begin, we construct the Laplacian matrix $\mathbf{L}$ using the adjacency matrix $\mathbf{W}$ and the degree matrix $\mathbf{D}$:

$$\mathbf{L} = \mathbf{D} - \mathbf{W}, \tag{3.5}$$

$$\mathbf{D}_{ii} = \sum_{j=1}^{n} \mathbf{W}_{ij}. \tag{3.6}$$

The Laplacian matrix $\mathbf{L}$ possesses a spectrum comprising n eigenvalues λ_0, λ_1, $\cdots$, λ_{n-1}, where $\lambda_0 = 0$ and $\lambda_{i+1} > \lambda_i$. The associated eigenvectors, corresponding to these eigenvalues, can be viewed as indicator vectors that aid in partitioning the graph. Moreover, the magnitude of the eigenvalues is indicative of the cost associated with the corresponding cut [49]. Consequently, the eigenvectors associated with the smallest eigenvalues are those linked to partitions with the minimum cost.

By mapping each dataset class index to a node in the spectral clustering framework, we can effectively address the problem of dataset complexity assessment. The adjacency matrix $\mathbf{W}$ and Laplacian matrix $\mathbf{L}$ are both square matrices of size $n \times n$, where n represents the total number of classes in the dataset. The weight w_{ij} in the matrix $\mathbf{W}$ signifies the similarity between different classes. Therefore, a complex dataset characterized by significant overlap between classes will yield a Laplacian spectrum with larger eigenvalues. The magnitude of the eigenvalues in the Laplacian spectrum reflects the similarity between classes and can serve as a measure of dataset complexity.

3.2.4 Dataset complexity calculation

To ensure the symmetry of the similarity matrix $\mathbf{X}$ derived from the Monte Carlo method, we transform it into a symmetric similarity matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ using the Bray Curtis distance [50]:

$$\mathbf{W}_{ij} = 1 - \frac{\sum_{q=1}^{q=n} |\mathbf{X}_{iq} - \mathbf{X}_{jq}|}{\sum_{q=1}^{q=n} |\mathbf{X}_{iq} + \mathbf{X}_{jq}|}, \tag{3.7}$$

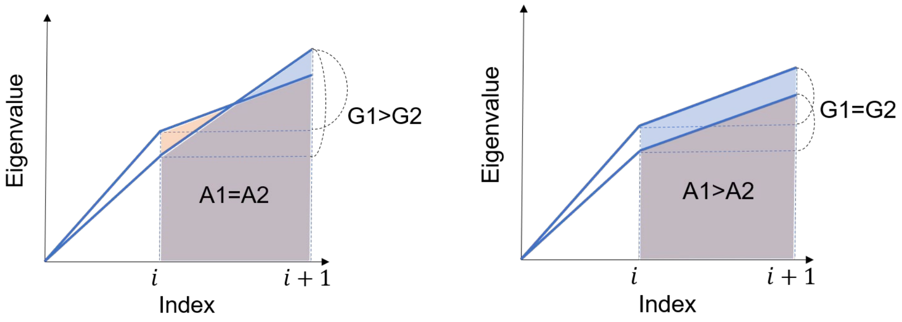
where $\mathbf{X}_i$ and $\mathbf{X}_j$ are the columns of the similarity matrix $\mathbf{X}$. The value $\mathbf{W}_{ij}$ represents the similarity between class i and class j. Using the symmetric adjacency matrix $\mathbf{W}$ and the degree matrix $\mathbf{D}$, we then construct the Laplacian matrix $\mathbf{L}$. The Laplacian matrix's spectrum consists of n eigenvalues $\lambda_0, \lambda_1, \cdots, \lambda_{n-1}$, where $\lambda_0 = 0$ and $\lambda_{i+1} > \lambda_i$. Considering the influence of both the Area Under Laplacian Spectrum (AULS) and the gradient between adjacent eigenvalues on the assessment performance, we propose a simple yet effective method called cmsAULS for evaluating dataset complexity. It is defined as follows:

$$\text{cmsAULS} = \sum_{i=0}^{n-2} \text{cummax}(\Delta\lambda)_i, \tag{3.8}$$

$$\Delta\lambda_i = \frac{\lambda_{i+1} - \lambda_i}{n - i} \times \frac{\lambda_{i+1} + \lambda_i}{2} = \frac{\lambda_{i+1}^2 - \lambda_i^2}{2(n - i)}, \tag{3.9}$$

where the function cummax represents the cumulative maximum value of a vector. A smaller value of cmsAULS indicates a smaller overlap between classes in the dataset, while a larger value indicates a higher degree of overlap. Importantly, the computational complexity of cmsAULS is solely dependent on the calculation of the $n \times n$ size matrix and can be expressed as an asymptotic time complexity of $O(M \cdot d^2 \cdot n^2)$, where M is the number of selected samples and d is the downscaled dimension.

The concept illustration for cmsAULS is depicted in Figure 3.2. In the extreme case shown in Figure 3.2-(a), where two datasets have the same AULS, the evaluation of dataset complexity should rely on the gradient of adjacent eigenvalues. On the other hand, when two datasets exhibit an equal gradient between specific adjacent eigenvalues, as illustrated in Figure 3.2-(b), the AULS becomes a more suitable measure for assessing dataset complexity. By considering



(a)

(b)

Figure 3.2: Concept of the proposed method.

both the gradient between adjacent eigenvalues and the AULS, the cmsAULS achieves improved assessment performance.

3.3 Experiments

In this section, we conducted several experiments to evaluate the effectiveness of cmsAULS. In subsection 3.3.1, we provide an overview of the datasets used in our experiments. Subsequently, in subsection 3.3.2, we compare cmsAULS with various benchmark and state-of-the-art methods to assess its performance. Furthermore, in subsection 3.3.3, we investigate the combination of pretrained DCNN feature extractors with cmsAULS to achieve a higher Pearson correlation. Next, in subsection 3.3.4, we visualize the interclass distances of different datasets to verify the effectiveness of the obtained similarity matrix. Finally, in subsection 3.3.5, we analyze the influence of different reduced dimensions on the performance of cmsAULS.

3.3.1 Datasets

To evaluate the performance of cmsAULS, we utilized six types of 10-class image classification datasets with varying levels of complexity, similar to those used in [3]. These datasets include the well-known mnist [51], svhn [52], and cifar10 [53]. NotMNIST [54] is a dataset similar to mnist but consists of alphabets extracted from publicly available fonts. Additionally, stl10 [55] is a cifar10-inspired dataset where each class has fewer labeled training examples compared to cifar10, and the images are larger in size (96 × 96). Finally, compcars [56] is a dataset comprising 163 car makes with 1,716 car models. For our experiments, we selected the 10 most frequent car makes and resized the images to 128 × 128, resulting in 500 samples per class.

3.3.2 Comparison with benchmark and the state-of-the-art methods

In this section, we evaluate the performance of cmsAULS by comparing it with several benchmarks and state-of-the-art methods. To validate our approach in the dimension reduction phase, we employ different techniques such as CNN autoencoder, t-SNE, and their combination. We set the dimensions of the downscaled image feature to 128 and 3 using CNN autoencoder and t-SNE, respectively. In the matrix construction phase, we carefully select the hyperparameters M, E, and k, which are set to 100, 100, and 3, respectively. These choices effectively contribute to calculating the complexity of the dataset. To assess the validity of cmsAULS, we compare it with 10 different descriptors [25,28], CSG [3], and the AULS method. Furthermore, we evaluate the performance of these methods by computing the Pearson correlation and p-value between the error rates of three DCNN models (AlexNet [9], ResNet50 [10], and Xception [57]) and the complexity of the dataset. This assessment allows us to determine the effectiveness of the proposed methods in capturing the relationship between error rates and dataset complexity.

Table 3.1 presents the Pearson correlation and p-value between dataset complexity and the test error rates of the six 10-class datasets. The complexity is calculated using various methods, including N1, N2, N3, N4 (neighborhood methods), and cmsAULS. Among these methods, N1, N2, N3, and N4 perform better than other benchmark methods but fall short of achieving a Pearson correlation of 0.8. In contrast, cmsAULS significantly outperforms all other methods, with an average Pearson correlation of 0.96. These results demonstrate that the complexity calculated by cmsAULS exhibits a strong positive correlation with DCNN test error rates. We refer to a 9-layer CNN autoencoder as CAE, and Comb. stands for the combination of CNN autoencoder and t-SNE. Table 3.2 provides the network structure details of the 9-layer CNN autoencoder. The architecture includes Convolution (Conv) layers, MaxPooling (MaxPool) layers, and Transposed Convolution (TConv) layers. The number specified after Conv represents the kernel size used in the corresponding convolution layer.

Table 3.3 reports the test error rates of three DCNN models on the six 10-class datasets. To ensure fairness, we directly utilize the reported test error rates from [3]. Additionally, Table 3.4

	AlexNet			ResNet50			Xception		
Method	CAE	t-SNE	Comb.	CAE	t-SNE	Comb.	CAE	t-SNE	Comb.
F1	-0.575 (0.233)	-0.485 (0.329)	-0.522 (0.288)	-0.582 (0.225)	-0.458 (0.361)	-0.469 (0.348)	-0.543 (0.266)	-0.413 (0.416)	-0.440 (0.382)
F2	-0.357 (0.487)	-0.061 (0.908)	0.232 (0.659)	-0.370 (0.470)	-0.024 (0.964)	0.158 (0.765)	-0.317 (0.541)	-0.093 (0.862)	0.151 (0.775)
F3	-0.461 (0.357)	-0.262 (0.616)	-0.423 (0.403)	-0.424 (0.402)	-0.287 (0.582)	-0.365 (0.476)	-0.375 (0.464)	-0.207 (0.694)	-0.328 (0.526)
F4	0.229 (0.663)	-0.335 (0.516)	-0.417 (0.411)	0.186 (0.725)	-0.333 (0.519)	-0.356 (0.488)	0.276 (0.597)	-0.266 (0.610)	-0.317 (0.541)
N1	0.771 (0.073)	0.704 (0.119)	0.710 (0.114)	0.712 (0.112)	0.663 (0.151)	0.653 (0.160)	0.677 (0.140)	0.612 (0.196)	0.613 (0.196)
N2	0.683 (0.135)	0.688 (0.131)	0.778 (0.068)	0.634 (0.177)	0.647 (0.165)	0.680 (0.137)	0.590 (0.218)	0.592 (0.216)	0.667 (0.148)
N3	0.776 (0.070)	0.741 (0.092)	0.744 (0.090)	0.709 (0.115)	0.692 (0.127)	0.666 (0.148)	0.676 (0.140)	0.644 (0.167)	0.637 (0.174)
N4	0.269 (0.606)	0.552 (0.256)	0.581 (0.227)	0.133 (0.802)	0.525 (0.285)	0.537 (0.272)	0.130 (0.806)	0.473 (0.344)	0.497 (0.316)
T1	0.418 (0.410)	-0.563 (0.244)	0.209 (0.691)	0.356 (0.488)	-0.720 (0.107)	0.092 (0.862)	0.341 (0.508)	-0.635 (0.175)	0.101 (0.849)
T2	-0.774 (0.071)	-0.774 (0.071)	-0.774 (0.071)	-0.746 (0.088)	-0.746 (0.088)	-0.746 (0.088)	-0.785 (0.065)	-0.785 (0.065)	-0.785 (0.065)
AULS	0.722 (0.105)	0.921 (0.009)	0.933 (0.006)	0.722 (0.105)	0.911 (0.012)	0.907 (0.012)	0.665 (0.150)	0.895 (0.016)	0.888 (0.018)
CSG	0.690 (0.129)	0.908 (0.012)	0.900 (0.015)	0.740 (0.093)	0.938 (0.006)	0.943 (0.005)	0.676 (0.140)	0.914 (0.011)	0.911 (0.011)
cmsAULS	0.753 (0.084)	0.962 (0.002)	**0.969 (0.001)**	0.838 (0.037)	0.955 (0.003)	**0.961 (0.002)**	0.784 (0.065)	0.941 (0.005)	**0.950 (0.004)**

Table 3.1: Pearson correlation and p-value between the complexity and the test error rates of the six 10-class datasets.

Table 3.2: The network structure of the 9-layer CNN autoencoder.

Layers	Operator	Resolution	Channels
1	Conv3 & MaxPool	32 × 32	64
2	Conv3 & MaxPool	16 × 16	128
3	Conv3 & MaxPool	8 × 8	256
4	Conv3 & MaxPool	4 × 4	256
5	Conv1	4 × 4	8
6	TConv2	8 × 8	128
7	TConv2	16 × 16	256
8	TConv2	32 × 32	512
9	TConv2	64 × 64	512

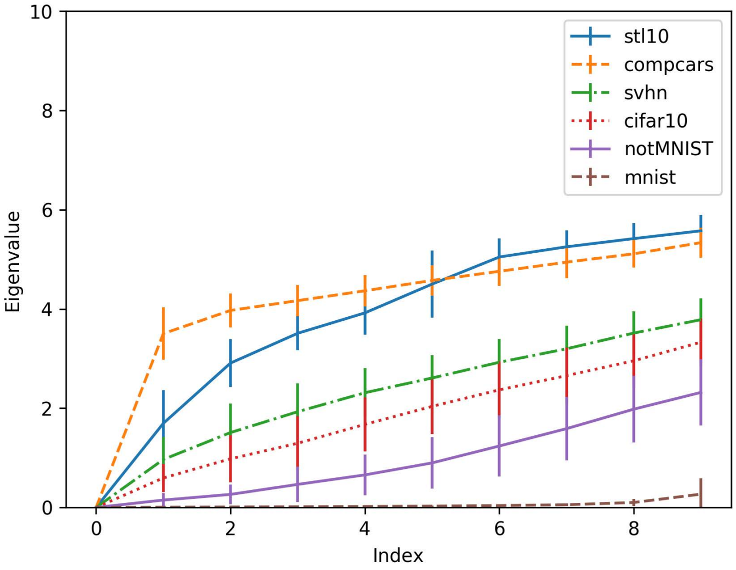


Figure 3.3: Laplacian spectrum of the six 10-class datasets (Comb.).

displays the calculated complexity scores for the six 10-class datasets. Notably, simpler datasets, such as mnist, exhibit lower complexity scores, whereas more complex datasets receive higher scores. Figure 3.3 visualizes the Laplacian spectrum of the six 10-class datasets. The figure highlights the trend that datasets with higher test error rates tend to have larger Laplacian spectra. This observation further supports the notion that dataset complexity influences the performance of DCNN models.

Table 3.3: Test error rates for three DCNN models on the six 10-class datasets [3].

Dataset	AlexNet	ResNet50	Xception
mnist	0.01	0.05	0.01
notMNIST	0.05	0.04	0.03
svhn	0.08	0.07	0.03
cifar10	0.18	0.19	0.06
stl10	0.69	0.63	0.69
compcars	0.70	0.88	0.86

Table 3.4: Complexity of the six 10-class datasets (Comb.).

Dataset	cmsAULS	CSG	AULS
mnist	0.144	0.045	0.675
notMNIST	0.693	0.747	9.294
svhn	1.100	1.826	20.142
cifar10	1.224	2.043	22.112
stl10	1.914	3.546	49.134
compcars	3.170	3.840	58.353

3.3.3 The effectiveness of pretrained DCNN feature extractors

In this section, we aim to improve the Pearson correlation by incorporating pretrained DCNN feature extractors with cmsAULS. Additionally, we evaluate the robustness of cmsAULS by calculating the Pearson correlation between complexity and test error rates for five out of the six 10-class datasets, with one dataset removed at a time. To leverage the powerful image classification capabilities of ImageNet, we utilize EfficientNet [58] models trained with Noisy Student [59] as our feature extractors. Specifically, we employ EfficientNet-B4 extractors that have demonstrated superior performance compared to other versions in our experiments. Noisy Student training is a semi-supervised learning approach that excels even when abundant labeled data is available, thereby enhancing the classification performance of supervised learning. Consequently, EfficientNet models trained with Noisy Student tend to yield better feature representations for images. Building upon the successful performance of t-SNE in previous experiments, we combine EfficientNet-B4 with t-SNE for dimensionality reduction of the extracted image features in this experiment. By employing t-SNE, we can effectively reduce the dimensionality while preserving the underlying structure and relationships within the data. This combined approach enables us to

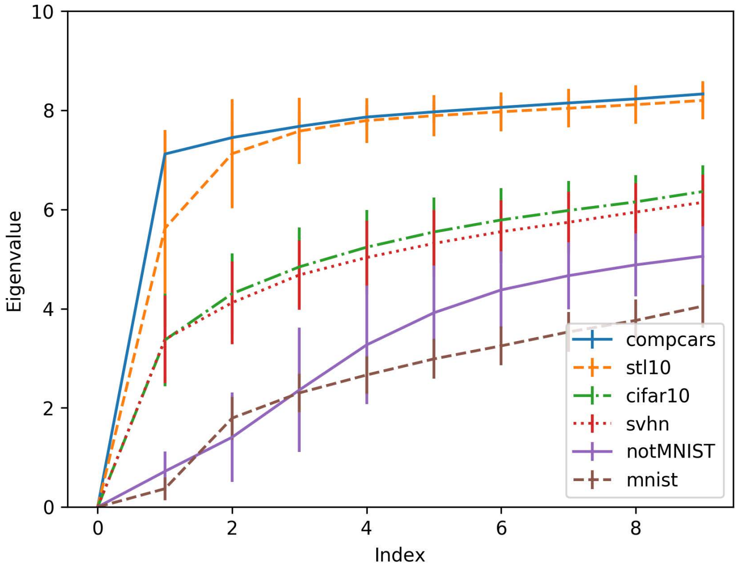


Figure 3.4: Laplacian spectrum of the six 10-class datasets (EfficientNet-B4 and t-SNE).

Table 3.5: Pearson correlation and p-value between the complexity and the test error rates of the six 10-class datasets.

Method	Evaluation	AlexNet	ResNet50	Xception
cmsAULS	Corr	**0.989**	**0.986**	**0.988**
cmsAULS	p-val	<**0.001**	<**0.001**	<**0.001**
CSG	Corr	0.956	0.965	0.948
CSG	p-val	0.003	0.002	0.004
AULS	Corr	0.942	0.913	0.898
AULS	p-val	0.005	0.011	0.015

obtain a compact and meaningful representation of the image features.

Table 3.5 presents the Pearson correlation and p-value between dataset complexity and test error rates for the six 10-class image datasets. It is evident from the table that the proposed method exhibits better correlation with all three DCNN models compared to CSG and AULS. Notably, cmsAULS achieves the lowest p-value (¡0.001), indicating reliable and statistically significant assessment results. Figure 3.4 visualizes the Laplacian spectrum of the six 10-class datasets, utilizing the combination of EfficientNet-B4 and t-SNE for dimensionality reduction of the image features. As observed in Figure 3.3, we confirm that datasets with higher test error rates tend to have larger Laplacian spectra. Table 3.6 demonstrates the Pearson correlation between dataset complexity and test error rates for the five 10-class datasets, with one dataset removed at a time.

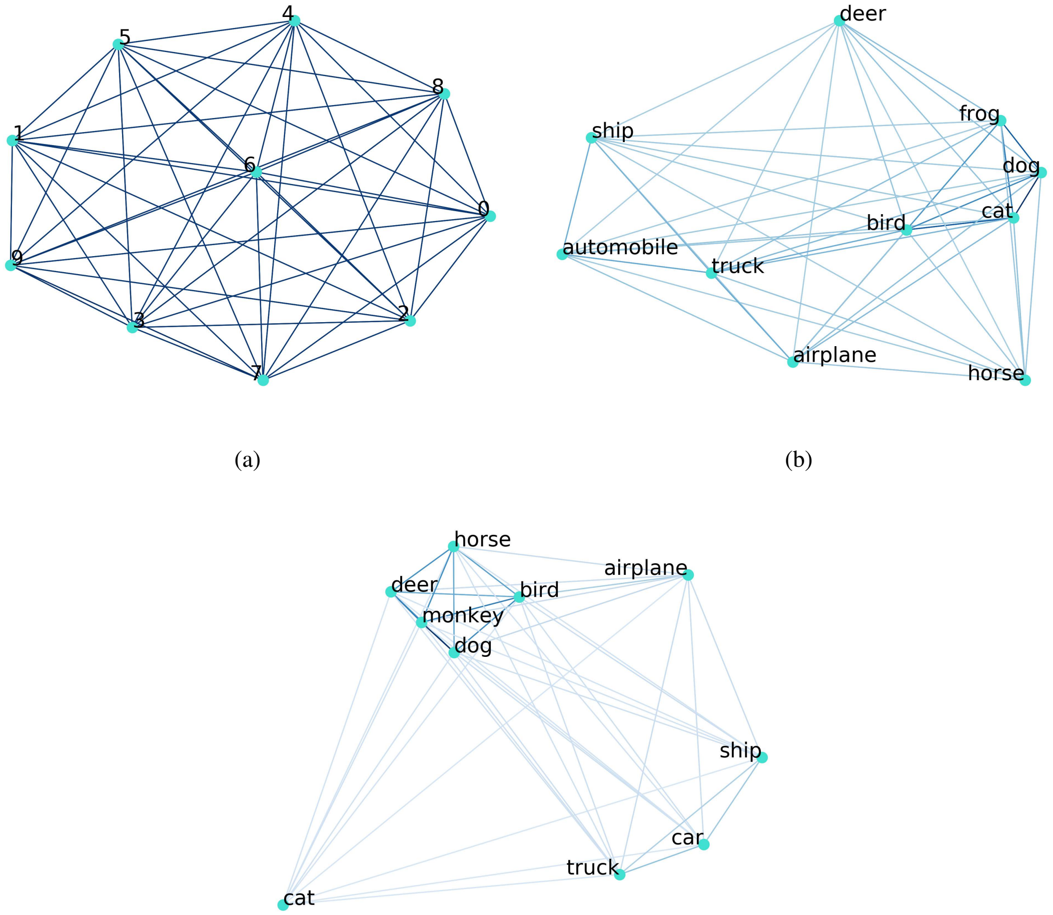
Table 3.6: Pearson correlation between the complexity and the test error rates of the five 10-class datasets (excluding one of the six datasets).

Remove	Method	AlexNet	ResNet50	Xception
mnist	**cmsAULS**	**0.988**	**0.992**	**0.996**
	CSG	0.952	0.978	0.960
	AULS	0.951	0.936	0.922
notMNIST	**cmsAULS**	**0.988**	**0.985**	**0.987**
	CSG	0.952	0.961	0.947
	AULS	0.936	0.900	0.892
svhn	**cmsAULS**	**0.992**	**0.991**	**0.991**
	CSG	0.976	0.989	0.968
	AULS	0.975	0.949	0.929
cifar10	**cmsAULS**	**0.989**	**0.987**	**0.991**
	CSG	0.957	0.967	0.962
	AULS	0.952	0.924	0.931
stl10	**cmsAULS**	**0.994**	**0.988**	**0.984**
	CSG	0.973	0.957	0.939
	AULS	0.921	0.893	0.859
compcars	**cmsAULS**	**0.992**	**0.980**	**0.976**
	CSG	0.937	0.924	0.887
	AULS	0.908	0.894	0.845

The results reaffirm the robust performance of cmsAULS in the task of dataset complexity assessment. Considering the results presented in Tables 3.5 and 3.6, we can ascertain the validity and robustness of cmsAULS. It is worth noting that the image features extracted by EfficientNet-B4 exhibit greater similarity to the tested DCNN models (AlexNet, ResNet50, and Xception), resulting in better performance compared to the CNN autoencoder.

3.3.4 Interclass distance visualization

In this section, we aim to visually assess the effectiveness of the obtained similarity matrix by examining the interclass distance within different datasets. By analyzing the interclass distance, we can validate the accuracy of the similarity matrix $\mathbf{W}_{ij}$ that we have generated. We have previously demonstrated a strong Pearson correlation between the dataset complexity, as calculated by our method, and the error rates of DCNN (Deep Convolutional Neural Network) models. However, we can further utilize the similarity matrix $\mathbf{W}_{ij}$ to visualize the interclass distance present



(a)

(b)

(c)

Figure 3.5: Interclass distance of different datasets: (a) mnist, (b) cifar10, and (c) stl10.

in a dataset. To accomplish this, we construct a dissimilarity matrix $\mathbf{U}_{ij} = 1 - \mathbf{W}_{ij}$. This dissimilarity matrix reflects the dissimilarities or distances between different classes within the dataset. Next, we employ multidimensional scaling (MDS) techniques to project the dataset's interclass distances onto a two-dimensional space.

In this section, we aim to visually verify the effectiveness of the obtained similarity matrix by visualizing the interclass distance of different datasets. While we have already demonstrated the high Pearson correlation between the dataset complexity calculated by our method and DCNN test error rates, we can further utilize the similarity matrix $\mathbf{W}_{ij}$ to visualize the interclass distance within a dataset. To achieve this, we construct a dissimilarity matrix $\mathbf{U}_{ij}$, which is computed as the complement of the similarity matrix, i.e., $\mathbf{U}_{ij} = 1 - \mathbf{W}_{ij}$. This dissimilarity matrix captures the pairwise distances between classes in the dataset. We then apply multidimensional scaling (MDS) to reduce the dimensionality of the dissimilarity matrix to two dimensions. By visualizing the interclass distance in a dataset using MDS, we gain insight into the arrangement and relationships between different classes. This visualization provides further evidence of the effectiveness of the obtained similarity matrix in capturing the intrinsic characteristics and structures of the dataset.

3.3.5 The influence of the image feature's dimension

In this section, we investigate the influence of different reduced dimensions on the performance of cmsAULS. We conduct experiments using two widely used dimension reduction methods, namely t-SNE and PCA, with varying reduced dimensions. For t-SNE, we set the reduced dimensions to 2 and 3, as they are commonly employed in visualization tasks. Additionally, we employ PCA with reduced dimensions of 3 and 50, along with contribution rates of 0.90 and 0.95. Table 3.7 presents the experimental results. It is evident from the table that when the reduced dimension is small (e.g., three dimensions), t-SNE achieves the best performance and outperforms PCA by a significant margin. However, when using PCA with a reduced dimension set to 3 and a contribution rate of 0.90, our method still achieves good performance while demonstrating faster execution time. The results presented in Table 3.7 emphasize the importance of selecting appropriate dimension reduction methods and determining the optimal reduced dimension for the cmsAULS approach.

Table 3.7: Pearson correlation and p-value between the complexity and the test error rates of the six 10-class datasets with different reduced dimensions.

Method	Evaluation	AlexNet	ResNet50	Xception
t-SNE (2*d*)	Corr	0.511	0.402	0.401
t-SNE (2*d*)	p-val	0.300	0.430	0.431
t-SNE (3*d*)	Corr	**0.969**	**0.961**	**0.950**
t-SNE (3*d*)	p-val	**0.001**	**0.002**	**0.004**
PCA (3*d*)	Corr	0.291	0.362	0.298
PCA (3*d*)	p-val	0.575	0.481	0.567
PCA (50*d*)	Corr	0.784	0.877	0.813
PCA (50*d*)	p-val	0.065	0.022	0.049
PCA (0.90)	Corr	0.796	0.887	0.825
PCA (0.90)	p-val	0.058	0.019	0.043
PCA (0.95)	Corr	0.774	0.873	0.808
PCA (0.95)	p-val	0.070	0.023	0.052

3.4 Conclusion

In this chapter, we introduce a novel method called cmsAULS, which aims to enhance the assessment performance of image dataset complexity. We leverage the concept of Laplacian spectrum size, derived from spectral clustering theory, as an indicator of class similarities within a dataset, thereby enabling the assessment of dataset complexity. Our method focuses on two key factors that impact the Laplacian spectrum size: the Average Unweighted Laplacian Similarity (AULS) and the gradient between adjacent eigenvalues. By emphasizing these elements, our method achieves superior assessment performance compared to existing methods in the field. As a result of our approach, we surpass the performance of state-of-the-art methods in the assessment of dataset complexity across six different datasets. These findings highlight the effectiveness and superiority of our proposed cmsAULS method in accurately evaluating the complexity of image datasets.

Chapter 4

Compressed Gastric Data Generation Based on Soft-Label Dataset Distillation

4.1 Introduction

This chapter introduces a novel approach for generating compressed gastric images using a technique called soft-label dataset distillation. The primary objective is to facilitate the sharing of medical data across different agencies and enable the development of highly accurate computer-aided diagnosis (CAD) systems. Sharing medical data is crucial for seamless healthcare information exchange, but it poses challenges due to the large sizes of medical datasets, the substantial memory requirements of deep convolutional neural network (DCNN) models, and the need for patients' privacy protection. These factors can impede the efficiency of medical data sharing. To address these challenges, the proposed method focuses on distilling essential information from medical image data and generating multiple compressed images that exhibit different data distributions. This approach ensures anonymity in medical data sharing. Moreover, the method incorporates a mechanism to extract significant parameters from DCNN models. By doing so, it reduces the memory footprint required to store trained models, thereby enhancing the efficiency of medical data sharing. The experimental results demonstrate the effectiveness of the proposed method. It not only compresses an entire gastric image dataset into multiple soft-label images but also significantly reduces the size of trained models to a fraction of their original size. The proposed method has valuable implications for the sharing of medical data, enabling

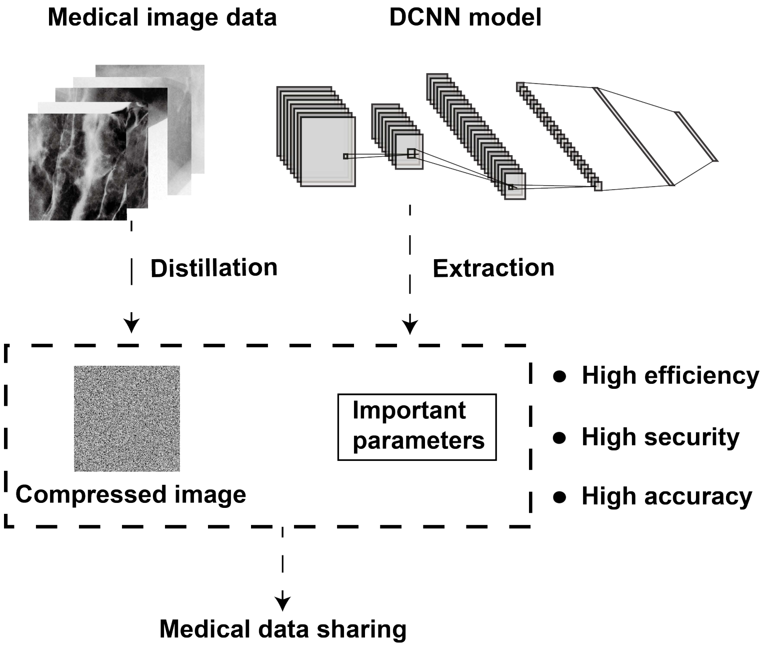


Figure 4.1: Overview of the proposed method.

efficient storage and transmission.

4.2 Method

Figure 4.1 provides an overview of the proposed method, which will be discussed in this section. The first step is the training data preprocessing procedure, explained in section 4.2.1. This section outlines the steps involved in preparing the training data for the proposed method. Next, section 4.2.2 presents the details of the compressed gastric image generation algorithm. This algorithm elaborates on how the compressed images with different data distributions are generated, following the soft-label dataset distillation approach. Finally, in section 4.2.3, the full gastric image classification performance using gastric patches is explained. This section describes how the proposed method can be applied to test the classification performance of full gastric images by using gastric patches.

4.2.1 Training data preprocessing

In this section, we propose a method to preprocess the training data while taking into account clinical settings. The full gastric X-ray images used in this research are depicted in Figure 4.1.

Figure 4.1(a) represents an example without gastritis (referred to as non-gastritis), while Figure 4.1(b) depicts an example with gastritis. Upon observing Figure 4.1, it becomes evident that a stomach without gastritis exhibits straight folds and a uniform mucosal surface pattern. Conversely, a stomach with gastritis displays non-straight folds and coarse mucosal surface patterns. Our dataset comprises gastric X-ray images with high resolutions, typically 2,048 × 2,048 pixels. In practical medical applications, working with high-resolution images can result in expensive computing costs. To address this issue, we adopt a patch-based detection/classification method. This approach enables effective utilization of pathology region and location information while eliminating the need for costly computations. Following our previous works [60], we divide each gastric X-ray image into patches for generating compressed gastric images.

To begin with, we partition each gastric X-ray image into multiple patches. Let $X_{\text{train}} \in \mathbb{R}^{d \times d}$ represent a full gastric X-ray image in the training data. The corresponding label for X_{train} is denoted as $Y_{\text{train}} \in 0, 1$, where $Y_{\text{train}} = 0$ indicates non-gastritis and $Y_{\text{train}} = 1$ indicates gastritis. Specifically, the full gastric images are divided into $H \times W$ patches, where H and W represent the number of patches in the vertical and horizontal directions, respectively. We define the patch-based dataset as $(\mathbf{x}, \mathbf{y}) = \{x_g, y_g\}_{g=1}^{G}$, where G denotes the number of patch images, $x_g \in \mathbb{R}^{d' \times d'}$ represents an image patch and y_g represents its corresponding label. We further annotate the patch images into three categories: $\mathcal{I}$, $\mathcal{N}$, and $\mathcal{P}$:

- $\mathcal{I}$: These patches are considered irrelevant as they lie outside the stomach region,
- $\mathcal{N}$: These patches are extracted from X-ray images without gastritis (non-gastritis) and are located within the stomach region,
- $\mathcal{P}$: These patches are extracted from X-ray images with gastritis and are also located within the stomach region.

To ensure accurate annotation, a radiological technologist manually processed the stomach region annotations in this research.

4.2.2 Compressed gastric image generation

In this section, we will describe the process of generating compressed gastric images using the soft-label dataset distillation. The framework of our method differs from the conventional

Algorithm 1 Training phase

```
Input: $\theta$: the random initial weights of a DCNN model; $\alpha$: learning rate; $K$: batch size; $T$: training steps; $M$: the number of compressed images; $\tilde{\mathbf{y}}_0$: initial value for $\tilde{\mathbf{y}}$; $\tilde{\alpha}_0$: initial value for $\tilde{\alpha}$
Output: $\tilde{\mathbf{x}}$: compressed images; $\tilde{\mathbf{y}}$: distilled labels; $\tilde{\alpha}$: optimized learning rate; $\theta_{\text{bn}}$: batch normalization parameters
Initialize $\tilde{\mathbf{x}} = \{\tilde{x}_m\}_{m=1}^M$ randomly, $\tilde{\mathbf{y}} = \{\tilde{y}_m\}_{m=1}^M \leftarrow \tilde{\mathbf{y}}_0$, $\tilde{\alpha} \leftarrow \tilde{\alpha}_0$
for each training step $t = 1$ to $T$ do
    Get a minibatch of training data:
    $(\mathbf{x}_t, \mathbf{y}_t) = \{x_{t,k}, y_{t,k}\}_{k=1}^K$
    Compute optimized weights with a gradient descent method:
    $\theta_{\text{opt}} \leftarrow \theta - \tilde{\alpha}\, \nabla_\theta \ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta)$
    Evaluate the objective function on the minibatch of training data:
    $\mathcal{L} = \ell(\mathbf{x}_t, \mathbf{y}_t, \theta_{\text{opt}})$
    Update distilled data:
    $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} - \alpha \nabla_{\tilde{\mathbf{x}}} \mathcal{L}$, $\tilde{\mathbf{y}} \leftarrow \tilde{\mathbf{y}} - \alpha \nabla_{\tilde{\mathbf{y}}} \mathcal{L}$, and $\tilde{\alpha} \leftarrow \tilde{\alpha} - \alpha \nabla_{\tilde{\alpha}} \mathcal{L}$
    if the DCNN model has batch normalization layers then
        Save the batch normalization parameters as $\theta_{\text{bn}}$
    end if
end for
```

neural network training and testing phases. Therefore, we will provide a brief overview of our approach. In the training phase, our objective is to distill the information from a large dataset into several compressed and anonymous images. This is achieved by utilizing a twice-differentiable loss function and updating the images through gradient descent. The aim is to minimize the loss function and extract the essential information from the dataset. In the test phase, we utilize the optimized distilled images to evaluate the accuracy achieved during the training phase. These images serve as a representation of the original dataset and are used to assess the effectiveness of the training process.

Algorithm 1 presents the training phase of our approach. The input and output settings for this phase are described below. In the training phase, we initialize the weights of a random DCNN model as θ. The learning rate, batch size, and the number of training steps are denoted by α, K, and T, respectively. The total number of compressed images is represented by M. Additionally, we have the initial value of distilled labels, $\tilde{\mathbf{y}}_0$, and the initial value of the optimized learning rate, $\tilde{\alpha}_0$. During the training phase, we obtain several outputs that are essential for the test phase. These outputs include the compressed images $\tilde{\mathbf{x}}$, the distilled labels $\tilde{\mathbf{y}}$, the optimized learning rate

$\tilde{\alpha}$, and the batch normalization parameters θ_{bn}. These outputs play a crucial role in evaluating the performance of the trained model during the test phase.

We next show the details of the algorithm for generating compressed gastric images. In this algorithm, we use a patch-based gastric training set denoted as $(\mathbf{x}, \mathbf{y}) = \{x_g, y_g\}_{g=1}^{G}$, where G represents the number of training images. The variables x_g and y_g correspond to the gastric image and its corresponding label, respectively. To parameterize the weights of a random DCNN model, we use θ. Additionally, we define a twice-differentiable loss function, $\ell(\mathbf{x}, \mathbf{y}, \theta)$, which represents the loss of the DCNN model on the entire training set $(\mathbf{x}, \mathbf{y})$. In our compressed gastric image generation method, we aim to distill valuable information from the entire training set $(\mathbf{x}, \mathbf{y})$ into a significantly smaller distilled dataset, denoted as $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) = \{\tilde{x}_m, \tilde{y}_m\}_{m=1}^{M}$. Here, M represents the number of compressed images, which is much smaller than G ($M \ll G$). The variables $\tilde{x}_m$ and $\tilde{y}_m$ correspond to the distilled image and its corresponding distilled label, respectively.

In our compressed gastric image generation algorithm, we assign soft labels $\tilde{\mathbf{y}}$ to the compressed images $\tilde{\mathbf{x}}$. These soft labels can be represented as probability distributions over different categories, such as $\mathcal{I}$, $\mathcal{N}$, and $\mathcal{P}$, following the approach described by Hinton et al. in the concept of distillation [61]. Since the compressed images $\tilde{\mathbf{x}}$ are not samples from the actual distribution, we can have a significantly smaller number of compressed images compared to the original training set. This compression allows us to capture the common features shared across different categories of gastric patches effectively. By incorporating soft labels into the training process, we introduce a form of regularization, which can improve the classification performance compared to the original dataset distillation method [35]. This regularization helps to generalize the learned knowledge and reduce overfitting. Moreover, it is possible to compress the entire gastric training set into just one compressed image with soft labels, achieving maximum compression while still retaining the essential information. During the distilling process, the optimized weights are computed according to the following equation:

$$\theta_{\text{opt}} \leftarrow \theta - \tilde{\alpha} \nabla_{\theta} \ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta), \tag{4.1}$$

where θ_{opt} represents the optimized weights of the DCNN model. θ denotes the initial weights, and $\tilde{\alpha}$ refers to the optimized learning rate. The loss function $\ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta)$ quantifies the discrepancy between the predictions outputted by the DCNN model with weights θ and the ground truth labels

$\tilde{\mathbf{y}}$ for the compressed dataset $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$. This loss function should be twice-differentiable to enable efficient optimization.

The objective function for our compressed gastric image generation method can be defined as follows:

$$\theta^* = \arg\min \ell(\mathbf{x}, \mathbf{y}, \theta). \tag{4.2}$$

Contrary to the general training goal of DCNNs, which aims to find the optimal parameters θ^*, our objective is to find the optimal compressed images $\tilde{\mathbf{x}}^*$, distilled labels $\tilde{\mathbf{y}}^*$, and optimized learning rate $\tilde{\alpha}^*$. These variables represent the compressed representation of the training set that leads to the minimum empirical error when used with the derived weights θ_{opt}.The objective function can be defined as:

$$\begin{aligned} \tilde{\mathbf{x}}^*, \tilde{\mathbf{y}}^*, \tilde{\alpha}^* &= \arg\min \mathcal{L}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\alpha}; \theta), \\ &= \arg\min \ell(\mathbf{x}, \mathbf{y}, \theta_{\text{opt}}), \\ &= \arg\min \ell(\mathbf{x}, \mathbf{y}, \theta - \tilde{\alpha} \nabla_\theta \ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta)), \end{aligned} \tag{4.3}$$

where $\ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta)$ is twice-differentiable, and $\mathcal{L}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\alpha}; \theta)$ is differentiable.

To obtain the optimal compressed images, distilled labels, and optimized learning rate, we update the compressed images $\tilde{\mathbf{x}}$, distilled labels $\tilde{\mathbf{y}}$ and optimized learning rate $\tilde{\alpha}$ at each distilling step with a gradient descent method as follows:

$$\begin{aligned} \tilde{\mathbf{x}} &\leftarrow \tilde{\mathbf{x}} - \alpha \nabla_{\tilde{\mathbf{x}}} \mathcal{L}, \\ \tilde{\mathbf{y}} &\leftarrow \tilde{\mathbf{y}} - \alpha \nabla_{\tilde{\mathbf{y}}} \mathcal{L}, \\ \tilde{\alpha} &\leftarrow \tilde{\alpha} - \alpha \nabla_{\tilde{\alpha}} \mathcal{L}, \end{aligned} \tag{4.4}$$

where $\nabla_{\tilde{\mathbf{x}}} \mathcal{L}$, $\nabla_{\tilde{\mathbf{y}}} \mathcal{L}$ and $\nabla_{\tilde{\alpha}} \mathcal{L}$ denote the gradients of $\mathcal{L}$ based on $\tilde{\mathbf{x}}$, $\tilde{\mathbf{y}}$ and $\tilde{\alpha}$, respectively, and α denotes the learning rate.

Next, we will describe the training process of the proposed algorithm. Initially, the compressed images, denoted as $\tilde{\mathbf{x}}$, are randomly initialized. The distilled labels, denoted as $\tilde{\mathbf{y}}$, and the optimized learning rate, denoted as $\tilde{\alpha}$, are initialized with $\tilde{\mathbf{y}}_0$ and $\tilde{\alpha}_0$, respectively. At each training step t, a minibatch of training data $(\mathbf{x}_t, \mathbf{y}_t)$ of size K is obtained. The distillation process involves computing optimized weights. To enhance the distillation results, the distillation process can be

Algorithm 2 Test phase

```
Input: $\theta$: the random initial weights of a DCNN model; $\tilde{\mathbf{x}}$: compressed images; $\tilde{\mathbf{y}}$: distilled labels; $\tilde{\alpha}$: optimized learning rate; $\theta_{\text{bn}}$: batch normalization parameters
Output: Pred: predicted labels
if the DCNN model does not have batch normalization layers then
    Compute optimized weights with the distilled data:
    $\theta_{\text{opt}} \leftarrow \theta - \tilde{\alpha}\, \nabla_{\theta}\ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta)$
else
    Compute optimized weights with the distilled data and batch normalization parameters:
    $\theta_{\text{opt}} \leftarrow \theta - \tilde{\alpha}\, \nabla_{\theta}\ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta_{\text{bn}}, \theta)$
end if
Predict the labels of the test data:
Pred = model $(\mathbf{x}_{\text{test}}, \mathbf{y}_{\text{test}}, \theta_{\text{opt}})$
```

extended by performing multiple distill epochs and multiple distill steps. This involves computing optimized weights through sequential gradient descent steps on the distilled dataset, repeated over a few epochs [62]. The sequential gradient descent steps can be defined as follows:

$$\theta_{i+1} \leftarrow \theta_i - \tilde{\alpha}\nabla_{\theta_i}\ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta_i), \tag{4.5}$$

where i denotes the distill steps. The objective function is evaluated on the minibatch of training data. The distilled data $\tilde{\mathbf{x}}$, $\tilde{\mathbf{y}}$, and $\tilde{\alpha}$ are updated based on a gradient descent method. Finally, if the DCNN model includes batch normalization layers in its architecture, the batch normalization parameters are saved as θ_{bn}.

Please note that the original dataset distillation method was designed for simple datasets such as MNIST and CIFAR10, using simple networks that do not include batch normalization layers. In the presence of batch normalization layers in a deep convolutional neural network (DCNN) model, the mean and variance information of batches are treated as constant and assumed to remain unchanged during the gradient steps in the distillation process. As a result, the information of batches cannot be distilled into the compressed images. However, it is sufficient to save only the batch normalization parameters θ_{bn} to reproduce the classification performance during the training phase. This feature is utilized in our method to reduce the sizes of the trained models.

Algorithm 2 shows the test phase of our method. During the test phase, the procedure for computing optimized weights and predicting labels depends on whether the DCNN model has batch normalization layers or not. If the DCNN model does not have batch normalization layers, we utilize the saved compressed images $\tilde{\mathbf{x}}$, distilled labels $\tilde{\mathbf{y}}$, and optimized learning rate $\tilde{\alpha}$ to compute the optimized weights θ_{opt} as follows:

$$\theta_{\mathrm{opt}} \leftarrow \theta - \tilde{\alpha}\,\nabla_{\theta}\ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta). \tag{4.6}$$

On the other hand, if the DCNN model includes batch normalization layers, we can compute the optimized weights using the distilled data and batch normalization parameters as follows:

$$\theta_{\mathrm{opt}} \leftarrow \theta - \tilde{\alpha}\,\nabla_{\theta}\ell(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta_{\mathrm{bn}}, \theta). \tag{4.7}$$

Once we obtain the optimized weights θ_{opt}, we can use the trained model to predict the labels on the test data $(\mathbf{x}_{\mathrm{test}}, \mathbf{y}_{\mathrm{test}})$ as follows:

$$\mathrm{Pred} = \mathrm{model}\,(\mathbf{x}_{\mathrm{test}}, \mathbf{y}_{\mathrm{test}}, \theta_{\mathrm{opt}}), \tag{4.8}$$

where Pred represents the predicted labels of the test data and can be used for the final full gastric image classification.

4.2.3 Full gastric image classification

In this section, we will explain the process of estimating the label of a full gastric X-ray image based on patches. First, when we have a test gastric image $X_{\mathrm{test}} \in \mathbb{R}^{d \times d}$, we divide it into $H \times W$ patches, following the same procedure as that used for training data. These divided patches are then inputted into a DCNN model with the optimized weights θ_{opt}, allowing us to obtain predicted labels (denoted as Pred) for each patch. Next, we calculate the numbers of patches for which the predicted labels are in the categories $\mathcal{N}$ (non-gastritis) and $\mathcal{P}$ (gastritis), denoted as Num($\mathcal{N}$) and Num($\mathcal{P}$), respectively. Since the patches extracted from outside the stomach (denoted as $\mathcal{I}$) are not relevant to the gastritis/non-gastritis prediction, they are not considered in the probability calculation. Finally, we estimate the label of the full gastric X-ray image as follows:

$$Y_{\mathrm{test}} = \begin{cases} 1 & \text{if } \frac{\mathrm{Num}(\mathcal{P})}{\mathrm{Num}(\mathcal{N}) + \mathrm{Num}(\mathcal{P})} \geq \delta \\ 0 & \text{otherwise} \end{cases}, \tag{4.9}$$

where δ is a threshold. Note that if $Y_{\mathrm{test}} = 1$, the estimation result of the full gastric X-ray image is gastritis, and if $Y_{\mathrm{test}} = 0$, the estimation result is non-gastritis.

4.3 Experiments

In this section, we will present the results of three experiments to demonstrate the effectiveness of our proposed method. In Section 4.3.1, we provide an overview of the experimental settings used in our method. The effectiveness of dataset reduction is evaluated and presented in Section 4.3.2. Furthermore, we demonstrate the effectiveness of model compression achieved by our method in Section 4.3.3. Lastly, in Section 4.3.4, we show the minimum number of compressed images required for different DCNN models as a measure of the efficiency of our approach.

4.3.1 Experimental settings

In our research, we utilized a medical dataset comprising gastric X-ray images obtained from 815 patients. Among these images, 240 were diagnosed as gastritis, while 575 were classified as non-gastritis cases. The ground truth labels (gastritis/non-gastritis) for each image were determined based on the results of patient diagnoses from endoscopic and X-ray examinations. The gastric X-ray images in the dataset had dimensions of 2,048 × 2,048 pixels and were grayscale images. For the training phase, we utilized a subset of the dataset containing images from 200 patients, with an equal distribution of 100 gastritis and 100 non-gastritis images. The remaining images from the dataset, comprising 140 gastritis and 475 non-gastritis images, were used as the test data for evaluating the performance of our proposed method.

In the data preprocessing stage, we divided all the gastric X-ray images into multiple patches with a size of 299 × 299 pixels and a sliding interval of 50 pixels. This patch size and sliding interval were determined through experimental evaluation. For the training data, these patches were labeled as $\mathcal{I}$, $\mathcal{N}$, or $\mathcal{P}$ by a radiological technologist. A patch was labeled as $\mathcal{I}$ if the regions inside the stomach accounted for less than 1% of the patch. If the regions inside the stomach accounted for more than 85% of the patch, it was labeled as either $\mathcal{N}$ (non-gastritis) or $\mathcal{P}$ (gastritis). The remaining patches were discarded. Consequently, we obtained training data consisting of patches labeled as $\mathcal{I}$, $\mathcal{N}$, and $\mathcal{P}$, with respective numbers of 48,385, 42,785, and 45,127 patches. During the training phase, these $\mathcal{I}$, $\mathcal{N}$, and $\mathcal{P}$ patches were used to train the DCNN models and generate compressed gastric images. For the test data, each of the remaining 615 gastric X-ray images was divided into 1,225 patches using the same procedure as the training data.

We conducted three experiments to evaluate the effectiveness of the proposed method. In all experiments, we used the compressed images that demonstrated the best classification performance for the patch-based training data. The performance evaluation was carried out on full gastric X-ray images from the test data. Throughout the experiments, we set the threshold δ to 0.4, as it tended to yield better classification performance. The random initial weights θ of all DCNN models utilized in our experiments were initialized using the default Xavier initializer. We employed the cross-entropy loss as the loss function. For evaluating the performance, we employed the following evaluation indexes: sensitivity (Sen), specificity (Spe), and the harmonic mean (HM) of Sen and Spe. The formulas for these indexes are as follows:

$$\mathrm{Sen} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{4.10}$$

$$\mathrm{Spe} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}, \tag{4.11}$$

$$\mathrm{HM} = \frac{2 \times \mathrm{Sen} \times \mathrm{Spe}}{\mathrm{Sen} + \mathrm{Spe}}, \tag{4.12}$$

where TP, TN, FP, and FN represent the numbers of true positive, true negative, false positive, and false negative, respectively. A higher sensitivity means maintaining a high ability to correctly identify positive cases, while decreasing specificity may lead to a higher number of false positives. Calculating the harmonic mean (HM) between sensitivity and specificity holds equal value as it balances the trade-off between these two metrics, providing a comprehensive evaluation of the detection performance.

4.3.2 Demonstration of the effectiveness of dataset reduction

In this section, we demonstrate the effectiveness of dataset reduction using the proposed method by comparing it with general network training. We used the ResNet18 [10] architecture for our experiments. First, we compared the dataset distillation approach with the original dataset distillation method [35]. We set the number of compressed images to 3 (one image per category) for soft-label dataset distillation (SLDD) and original dataset distillation (DD). Additionally, we used SLDD with only 1 distilled image. For SLDD (3), we initialized the soft labels

Table 4.1: Comparison of dataset distillation (ResNet18) and ResNet18.

Method	Sen	Spe	HM
SLDD (3)	0.886	0.869	**0.877**
DD (3)	0.829	0.884	0.856
ResNet18 (9000)	0.814	0.832	0.823
ResNet18 (6000)	0.907	0.760	0.827
ResNet18 (3000)	0.914	0.669	0.773
SLDD (1)	0.793	0.895	0.841

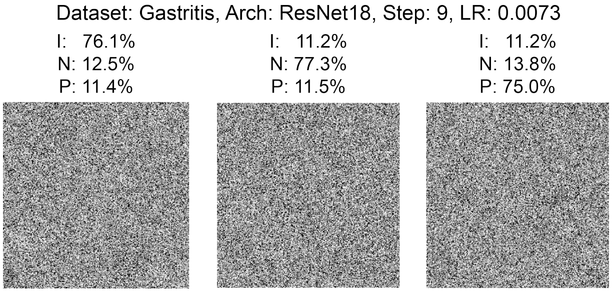


Figure 4.2: Compressed image generated in SLDD (3).

with one-hot values of the original labels ($\mathcal{I}$, $\mathcal{N}$, and $\mathcal{P}$). In SLDD (1), we initialized the soft label with the label $\mathcal{N}$, which tends to yield better classification performance. The distillation process consisted of 3 epochs with a total of 9 distillation steps. DD (3) had the same settings as SLDD (3) except for the fixed labels. During the training phase, we performed 400 epochs for SLDD (3), DD (3), and SLDD (1), and saved the distilled results for testing and evaluation. Since it is challenging for a DCNN model to learn from only a few images, we randomly selected 1,000, 2,000, and 3,000 images per category from the training data. We trained three ResNet18 models with these selected images until convergence, which served as the comparison methods in our experiments.

The test results are presented in Table 4.1. This table shows the classification performance of the proposed method and ResNet18 trained on random subsets for full gastric X-ray images. The ResNet18 model trained with 3,000 images per category (a total of 9,000 images) achieved an HM score of 0.823. In contrast, SLDD (1), which distilled all of the training data into a single compressed soft-label patch image for training, achieved an HM score of 0.841. Furthermore, SLDD (3), which distilled all of the training data into three compressed soft-label images for

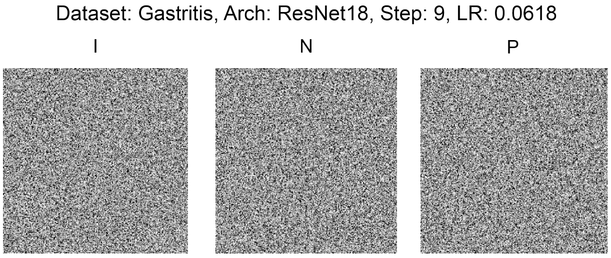


Figure 4.3: Compressed image generated in DD (3).

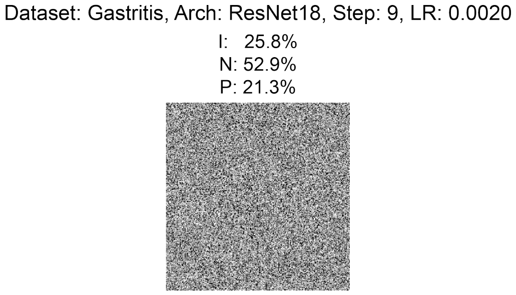


Figure 4.4: Compressed image generated in SLDD (1).

training, achieved an even higher HM score of 0.877 compared to DD (3). These results demonstrate that the proposed method exhibits high classification performance with only a few compressed gastric X-ray images, indicating the effectiveness of dataset reduction. Figures 4.2, 4.3 and 4.4 show examples of the compressed images used in our experiments. From these figures, it is evident that the gastric images have been completely anonymized. Consequently, the generated compressed patch images contain no private patient information, thereby facilitating privacy protection in the sharing of medical data.

4.3.3 Demonstration of the effectiveness of model compression

In this section, we investigate the model compression effectiveness of the proposed method by comparing different DCNN models with and without batch normalization layers. Typically, when a DCNN model includes batch normalization layers in its architecture, the information of each batch is stored in the parameters. During the training phase, the mean and variance of the batches are treated as constants and assumed to remain unchanged during the gradient steps. As a result, the information of the individual batches cannot be effectively distilled into compressed images. However, only the batch normalization parameters need to be saved in order to reproduce the classification performance of the training phase. This characteristic allows us to compress the

Table 4.2: Comparison of different models with batch normalization (bn) and without batch normalization (no_bn).

Model	Sen	Spe	HM
GoogLeNet (bn)	0.850	0.916	**0.882**
GoogLeNet (no_bn)	0.121	0.823	0.211
ResNet18 (bn)	0.836	0.905	**0.869**
ResNet18 (no_bn)	0.600	0.844	0.701
AlexNet (bn)	0.793	0.884	**0.836**
AlexNet (no_bn)	0.786	0.861	0.822
VGG16 (bn)	0.907	0.926	**0.916**
VGG16 (no_bn)	0.936	0.897	**0.916**

Table 4.3: Memory footprints of different models. Memory denotes saving all of the parameters of a model. Memory∗ denotes saving batch normalization parameters and distilled results.

Model	Memory	Memory∗	Compression rate
GoogLeNet	22.83MB	289.14KB	0.01266
ResNet18	42.64MB	250.23KB	0.00587
ResNet34	81.20MB	287.99KB	0.00354
AlexNet	217.44MB	201.29KB	0.00093
VGG16	512.21MB	402.01KB	0.00078
VGG19	532.46MB	402.01KB	0.00076

size of the model that needs to be stored.

In this experiment, we utilized different models, namely GoogLeNet [63], ResNet18 [10], AlexNet [9], and VGG16 [64], both with and without batch normalization layers. To begin, we compressed the images into three instances, with one image per category. We initialized the soft labels using one-hot encoding based on the original labels. The distillation epochs and steps were set to 1. During the training phase, we trained GoogLeNet, ResNet18, and AlexNet for 400 epochs, while VGG16 was trained for 200 epochs. After each epoch, we saved the distilled results and the batch normalization parameters for testing and evaluation purposes.

Table 4.4: Comparison of the minimum number of compressed images of different models.

Model	Sen	Spe	HM
GoogLeNet (1)	0.764	0.853	0.806
GoogLeNet (3)	0.850	0.916	**0.882**
ResNet18 (1)	0.800	0.855	0.827
ResNet18 (3)	0.836	0.905	**0.869**
ResNet34 (1)	0.800	0.926	0.858
ResNet34 (3)	0.893	0.899	**0.896**
AlexNet (1)	0.671	0.895	0.767
AlexNet (3)	0.793	0.884	**0.836**
VGG16 (1)	0.643	0.524	0.577
VGG16 (2)	0.921	0.926	**0.923**
VGG16 (3)	0.936	0.897	0.916
VGG19 (1)	0.614	0.891	0.727
VGG19 (2)	0.921	0.909	0.915
VGG19 (3)	0.921	0.933	**0.927**

4.3.4 Minimum number of compressed images

The test results are presented in Table 4.2, which demonstrates that models with batch normalization layers exhibit superior classification performance. Additionally, we conducted experiments using multiple models, and Table 4.3 displays the maximum compression rates achieved by different models. ”Memory” refers to the memory required to store all the parameters of the respective DCNN models, while ”Memory*” indicates the memory needed to store batch normalization parameters and distilled results. It is evident from Table 4.3 that our proposed method effectively reduces the memory size required to save trained models, highlighting the model compression effectiveness. For instance, VGG19 can achieve a maximum compression rate of 0.00076 by employing two compressed soft-label patch images. Moreover, we discovered that the minimum number of compressed images achievable varies across different models, and this characteristic is related to the maximum compression rate. Therefore, we will discuss the minimum number of compressed images for different models in the next section. It is worth noting that VGG16, for example, exhibits significantly lower test accuracy when the number of compressed images is set to 1 compared to when it is set to 3 (one image per class). In other words, VGG16 struggles to effectively distill pertinent information from the training data into a

Table 4.5: Parameters of different models. Parameter denotes the number of model parameters. Image denotes the minimum number of compressed images.

Model	Parameter	Image
GoogLeNet	5,984,915	1
ResNet18	11,176,963	1
ResNet34	21,285,123	1
AlexNet	57,000,643	1
VGG16	134,271,683	2
VGG19	139,581,379	2

single compressed image.

In this section, we utilized various models, including GoogLeNet, ResNet18, ResNet34, AlexNet, VGG16, and VGG19. We initially set the numbers of compressed images to the minimum values achievable by each model. For example, GoogLeNet had a minimum of 1 compressed image, while VGG16 had a minimum of 2 compressed images. As a comparison, we also set the numbers of compressed images to 3, corresponding to one image per category. For instance, GoogLeNet had 3 compressed images. The distillation epochs and steps were set to 1. During the training phase, we performed 400 epochs for GoogLeNet, ResNet18, and AlexNet, while ResNet34, VGG16, and VGG19 were trained for 200 epochs. After every epoch, we saved the distilled results and batch normalization parameters for subsequent testing and evaluation.

The test results are presented in Tables 4.4 and 4.5. From Table 4.4, it is evident that GoogLeNet, ResNet18, ResNet34, and AlexNet were successful in distilling valid information from the training data into only one compressed soft-label patch image. However, VGG16 and VGG19 were unable to effectively distill valid information into a single compressed patch image. As the compressed images were distilled using DCNN models, we believe that the minimum number of compressed images is influenced by the number of parameters in the models. Thus, we provide the parameter counts and minimum numbers of compressed images for different models in Table 4.5. The ”Parameter” column represents the number of parameters in each model, while the ”Image” column denotes the minimum number of compressed images achievable by the models. Table 4.5 demonstrates that there is a correlation between the minimum number of compressed images and the number of parameters in the models.

4.4 Conclusion

In this chapter, we introduced a novel method for generating compressed gastric images using soft-label dataset distillation. The aim of this method is to facilitate efficient and anonymous sharing of medical data. Our proposed approach not only compresses an entire medical dataset into a single compressed soft-label patch image but also reduces the size of the trained model to a fraction of its original size. This results in improved efficiency when sharing medical data. Importantly, the compressed images generated through distillation are completely anonymized and do not contain any private information about the patients. This significantly enhances the security of medical data sharing, ensuring patient privacy is maintained. Furthermore, our method achieves high classification performance even with a small number of compressed images. This demonstrates the effectiveness and efficiency of our approach in compressing and sharing medical data without compromising accuracy.

Chapter 5

Self-Supervised Transfer Learning for Automatic COVID-19 Detection

5.1 Introduction

In the context of the global COVID-19 pandemic, there is a growing need for computer-aided diagnosis systems that can quickly detect and triage COVID-19 cases using chest X-ray images. In this chapter, we propose a novel learning scheme called self-supervised transfer learning for COVID-19 detection from chest X-ray images. Recognizing that self-supervised learning alone may not provide sufficient representations for the target dataset, we introduce transfer learning from different datasets as a way to complement the limitations of self-supervised learning and improve representation learning. We demonstrate that knowledge learned from natural images through transfer learning can greatly benefit self-supervised learning on chest X-ray images, leading to enhanced representation learning performance for COVID-19 detection. Our method leverages the combination of transfer learning and self-supervised learning to acquire discriminative representations from chest X-ray images. Through extensive experiments, we achieve remarkable results on the largest available open COVID-19 chest X-ray dataset, with an HM score of 0.985, an AUC of 0.999, and an accuracy of 0.953. To enhance interpretability, we utilize the Grad-CAM++ visualization technique to generate visual explanations for different classes of chest X-ray images using our proposed method. This approach increases the interpretability of our model's predictions and provides insights into the learned representations.

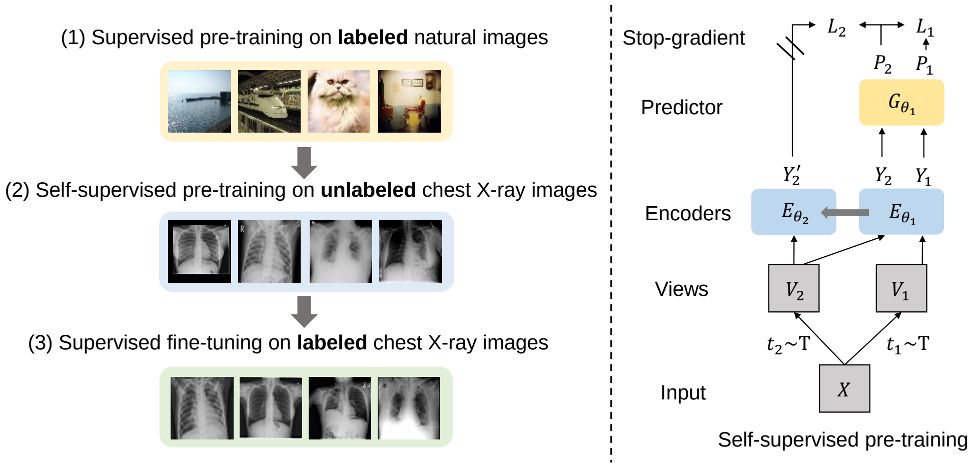

(1) Supervised pre-training on **labeled** natural images
(2) Self-supervised pre-training on **unlabeled** chest X-ray images
(3) Supervised fine-tuning on **labeled** chest X-ray images
Stop-gradient
Predictor
Encoders
Views
Input
L_2
L_1
P_2
P_1
G_{θ_1}
Y_2'
Y_2
Y_1
E_{θ_2}
E_{θ_1}
V_2
V_1
t_2~T
t_1~T
X
Self-supervised pre-training

Figure 5.1: Overview of the proposed method.

5.2 Method

To address the limited representations learned by self-supervised learning on the target dataset [65], we propose a method that combines transfer learning from different datasets with self-supervised learning to obtain more effective representations for COVID-19 detection from chest X-ray images. The core idea of our method is to leverage the knowledge learned from natural images through transfer learning, which can compensate for the shortcomings of self-supervised learning and enhance representation learning performance. The transfer learning is performed in the first stage of our method, where we conduct supervised pre-training on labeled natural images, such as the widely used ImageNet dataset [66]. This step allows our model to learn meaningful and discriminative representations from natural images. The second stage of our method focuses on self-supervised pre-training on unlabeled chest X-ray images. This process helps the model learn specific features and patterns relevant to COVID-19 detection in chest X-ray images while leveraging the general knowledge obtained from the transfer learning stage. Finally, in the third stage, we perform supervised fine-tuning on labeled chest X-ray images. This step further refines the learned representations specifically for COVID-19 detection tasks. By combining transfer learning from natural images and self-supervised learning on chest X-ray images, our method is capable of learning highly discriminative representations that are well-suited for final fine-tuning. An overview of our proposed method is depicted in Figure 5.1. This combination of transfer learning and self-supervised learning enables our method to achieve superior performance in COVID-19 detection from chest X-ray images.

To generate two views of the input chest X-ray image X, we apply two randomly sampled transformations, t_1 and t_2, from a distribution T. This generates two transformed views, $V_1 = t_1(X)$ and $V_2 = t_2(X)$, as described in previous works [40, 67]. These transformations incorporate standard data augmentation techniques such as cropping, resizing, flipping, and Gaussian blur [39, 68, 69]. By applying these transformations, we increase the diversity and variability of the views, which aids in learning robust representations. The encoders process the two views, resulting in output representations denoted as Y_1, Y_2, and Y_2'. These representations capture the learned features and characteristics of the input views. Additionally, we employ a predictor module, denoted as G_{θ_1}, to process the output representations Y_1 and Y_2 and generate two additional representations, P_1 and P_2. The predictor module is designed to introduce an asymmetry in the network structure, which helps prevent learning from collapsing and promotes better representation learning [41]. To compare the normalized representations from the two views of the same image, we define three loss functions: L_1, L_2, and L. These loss functions quantify the similarity between the representations and encourage the network to learn meaningful and discriminative features. However, the specific formulation of these losses is not provided in your excerpt. By incorporating these components and optimizing the defined loss functions, our self-supervised learning process aims to learn informative and discriminative representations from unlabeled chest X-ray images. These learned representations can then be utilized in subsequent stages, such as fine-tuning, to improve the performance of COVID-19 detection.

$$L_1 = ||\hat{P}_1 - \hat{P}_2||_2^2 = 2 - 2 \cdot \frac{\langle P_1, P_2 \rangle}{||P_1||_2 \cdot ||P_2||_2}, \tag{5.1}$$

$$L_2 = ||\hat{P}_2 - \hat{Y}_2'||_2^2 = 2 - 2 \cdot \frac{\left\langle P_2, Y_2' \right\rangle}{||P_2||_2 \cdot ||Y_2'||_2}, \tag{5.2}$$

$$L = L_1 + L_2, \tag{5.3}$$

where $\hat{P}_i = P_i/||P_i||_2$ and $\hat{Y}_i' = Y_i'/||Y_i'||_2$ denote the normalized representations of V_i $(i = 1, 2)$. Then we use the total loss L to update the parameters of the encoder E_{θ_1}

$$\theta_1 \leftarrow \text{Opt}(\theta_1, \nabla_{\theta_1} L, \alpha). \tag{5.4}$$

Table 5.1: Details of the COVID-19 chest X-ray dataset [9] used in our study. “C”: COVID-19, “L”: Lung Opacity,“N”: Normal, and “V”: Viral Pneumonia.

Class	Total	Training image	Test image
C	3,616	2,893	723
L	6,012	4,810	1,202
N	10,192	8,154	2,038
V	1,345	1,076	269

Table 5.2: Test results of COVID-19 detection.

Method	Sen	Spe	HM	AUC	Acc
Ours (Cross + Transfer)	**0.972±0.003**	**0.997±0.001**	**0.985±0.001**	**0.999±0.000**	**0.953±0.001**
Transfer	0.944±0.004	0.994±0.001	0.968±0.002	0.997±0.000	0.936±0.001
Cross	0.923±0.005	0.991±0.001	0.955±0.002	0.995±0.000	0.908±0.001
BYOL	0.895±0.005	0.987±0.001	0.939±0.003	0.991±0.000	0.894±0.001
SimSiam	0.794±0.013	0.972±0.002	0.874±0.007	0.972±0.000	0.849±0.001
SimCLR	0.778±0.006	0.965±0.002	0.862±0.003	0.996±0.000	0.876±0.001
PIRL-Jigsaw	0.685±0.014	0.973±0.003	0.804±0.009	0.954±0.000	0.821±0.001
PIRL-Rotation	0.760±0.009	0.962±0.002	0.849±0.005	0.960±0.001	0.817±0.001
From Scratch	0.665±0.013	0.954±0.003	0.783±0.008	0.935±0.001	0.774±0.002

Here, Opt represents the optimizer used, and α denotes the learning rate. The weights of E_{θ_2} are updated using an exponential moving average [70] of the weights of E_{θ_1}, as follows:

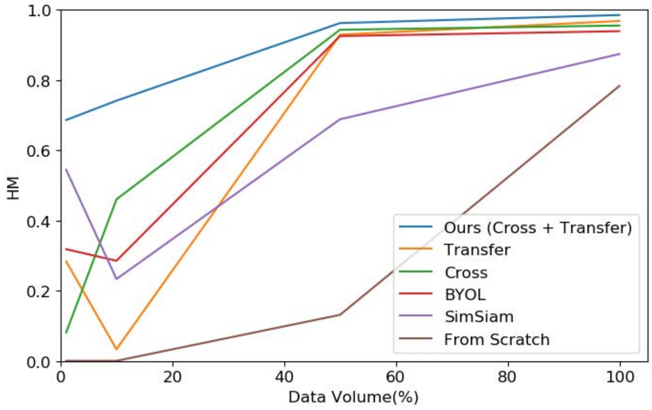
$$\theta_2 \leftarrow \tau\theta_2 + (1 - \tau)\theta_1. \tag{5.5}$$

The weight for θ_1 is determined by the parameter τ, which represents the degree of moving average. This update is performed after every iteration. To ensure stable training, the gradient is not back-propagated through the encoder E_{θ_2}, as discussed in [18]. By leveraging transfer learning techniques from natural images and applying self-supervised learning on chest X-ray images, we can acquire discriminative representations specifically tailored for chest X-ray images. Following the self-supervised learning process, the encoder E_{θ_1} is fine-tuned on labeled chest X-ray images to enhance its performance in detecting COVID-19.

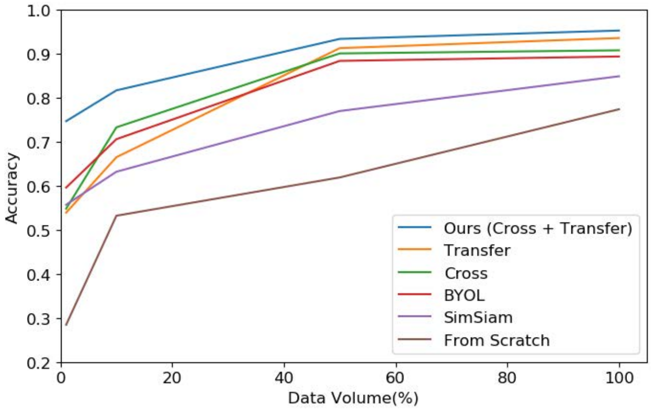
5.3 Experiments

5.3.1 Dataset and settings

The dataset utilized in our study is the largest publicly available COVID-19 chest X-ray dataset, as introduced by Rahman et al. [1]. Table 5.1 presents an overview of the dataset, which comprises a total of 21,165 grayscale chest X-ray images. The dataset is divided into four classes,



(a)



(b)

Figure 5.2: Test results of COVID-19 detection in different data volumes: (a) HM and (b) Acc.

namely COVID-19, Lung Opacity, Normal, and Viral Pneumonia. All the chest X-ray images have a resolution of 224 × 224 pixels. To create the training and test sets, we randomly selected 80% of the images for training and allocated the remaining 20% for testing. In our evaluation, we employed several metrics to assess the performance of our method. These included sensitivity (Sen), specificity (Spe), the harmonic mean (HM) of Sen and Spe, the area under the ROC curve (AUC), and the accuracy (Acc) in classifying the four classes. COVID-19 was considered the positive class, while the other classes were treated as negative.

The encoders used in our approach were based on the ResNet50 architecture proposed by He

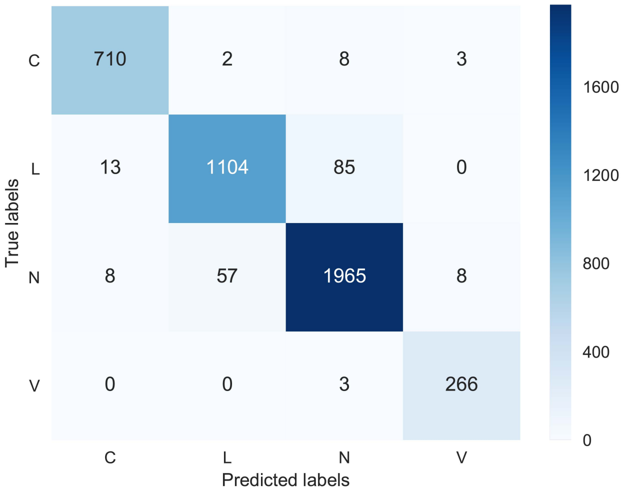


Figure 5.3: Confusion matrix for the best model of our method.

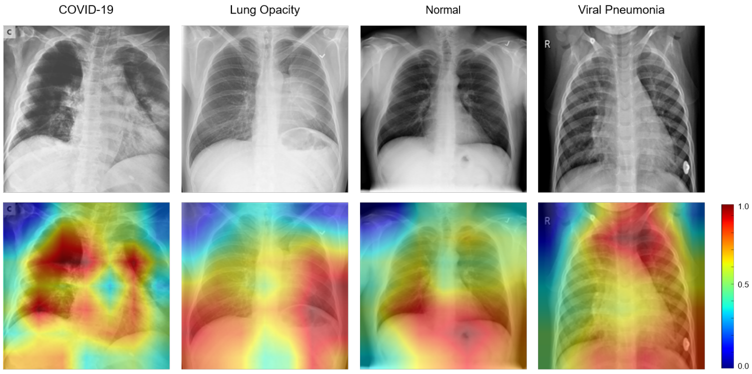


Figure 5.4: Grad-CAM++ visual explanations of the proposed method.

et al. [10]. Following the encoder, we employed a multilayer perceptron (MLP) consisting of a linear layer with an output size of 512, a batch normalization layer, a ReLU activation function, and another linear layer with an output size of 128. The predictor was implemented as an MLP, which included a linear layer with an output size of 4096, a batch normalization layer, a ReLU activation function, and a linear layer with an output size of 256. To optimize the model, we utilized an SGD optimizer with a learning rate α of 0.03, a momentum of 0.9, and a weight decay of 0.0004. The hyperparameter τ was set to 0.996, determining the degree of moving average during the parameter update process. We trained our model using a batch size of 256 and a generated view size of 112. For the self-supervised learning phase, we performed training for 40 epochs on the dataset. Subsequently, we fine-tuned the model on the dataset for an additional 30 epochs. During testing, we evaluated the model's performance using the average results from the last 10 epochs of fine-tuning.

To evaluate the effectiveness of our method in a low-shot data regime, we compared it with several state-of-the-art self-supervised learning methods, including Cross [71], BYOL [41], SimSiam [18], PIRL [38], and SimCLR [39]. We also considered transfer learning using ImageNet [66] pre-trained weights and training from scratch as comparative methods. It is worth noting that PIRL-Jigsaw and PIRL-Rotation are based on jigsaw and rotation pretext tasks, respectively, and were included in our evaluation. To simulate a low-shot data scenario, we randomly sampled objects from the dataset at varying proportions: 1%, 10%, and 50% of the total dataset size. These subsets were then used for the final fine-tuning process, allowing us to investigate the effectiveness of our method when only a limited amount of labeled data is available.

5.3.2 Experimental results

The test results for COVID-19 detection, using the entire training data, are presented in Table 5.2. The reported values represent the average and variance of the last 10 fine-tuning epochs. From the table, it is evident that our method significantly outperforms other comparative methods and demonstrates a substantial improvement in COVID-19 detection performance compared to using self-supervised learning or transfer learning alone. For instance, when utilizing all of the training data, the transfer learning approach achieves an HM score of 0.968, an AUC of 0.997, and an accuracy (Acc) of 0.936. Among the self-supervised learning methods, the best-performing technique, Cross [72], achieves an HM score of 0.955, an AUC of 0.995, and an Acc of 0.908. In contrast, our method, which combines transfer learning with self-supervised learning, achieves an HM score of 0.985, an AUC of 0.999, and an Acc of 0.953 on the largest open COVID-19 chest X-ray dataset. These results demonstrate that the knowledge learned from natural images through transfer learning contributes positively to self-supervised learning on chest X-ray images and enhances representation learning for COVID-19 detection. The test results for COVID-19 detection in a low-shot data regime are illustrated in Figure 5.2, representing the average performance from the last 10 fine-tuning epochs. Our method consistently outperforms other approaches across different data volumes, and even with only 50% of the training set, our method achieves promising detection performance.

Furthermore, Figure 5.3 presents the confusion matrix for the best model obtained using our method, achieving an HM score of 0.988, an AUC of 1.000, and an Acc of 0.956. The confusion matrix demonstrates the successful detection results achieved on the largest open COVID-19 chest X-ray dataset. To provide further insights into our method, we present examples of chest X-ray (CXR) images along with their Grad-CAM++ [73] visual explanations in Figure5.4. The highlighted regions in the visual explanations correspond to the decision-making process, increasing the confidence and reliability of our proposed method. These visualizations confirm the accuracy of decision-making by focusing on relevant regions within the CXR images [74].

5.4 Conclusion

In this study, we have introduced a novel learning scheme, termed self-supervised transfer learning, for COVID-19 detection using chest X-ray images. Through our experiments, we have demonstrated the efficacy of combining transfer learning from natural images with self-supervised learning on chest X-ray images, leading to improved representation learning for COVID-19 detection. By leveraging the knowledge acquired from transfer learning, our method effectively learns discriminative representations from chest X-ray images. This combination of transfer learning and self-supervised learning has yielded promising results on the largest open COVID-19 chest X-ray dataset, indicating its potential in enhancing the detection of COVID-19 cases. The implications of our method are significant in combating the transmission of COVID-19 and alleviating the burden on healthcare providers and radiologists. By enabling more accurate and efficient detection of COVID-19 cases from chest X-ray images, our approach can contribute to the timely identification and treatment of patients, ultimately aiding in the mitigation of the disease's impact.

Chapter 6

High-Accuracy Automatic COVID-19 Detection via Self-Supervised Learning and Batch Knowledge Ensembling

6.1 Introduction

In this chapter, we present a novel approach for detecting COVID-19 using chest X-Ray (CXR) images. COVID-19 and its variants have caused significant disruptions worldwide, impacting billions of lives in more than 200 countries and regions. CXR images have become a fast and convenient method for COVID-19 detection due to the common occurrence of radiological pneumonia findings in COVID-19 patients. Our method comprises two phases: self-supervised learning-based pretraining and batch knowledge ensembling-based fine-tuning. The self-supervised learning-based pretraining phase enables the learning of distinctive representations from CXR images without the need for manual annotations. In the fine-tuning phase, batch knowledge ensembling is introduced to utilize the category knowledge of images in a batch, enhancing the detection performance based on visual feature similarities. Unlike our previous implementation, the incorporation of batch knowledge ensembling into the fine-tuning phase reduces the memory usage in self-supervised learning while improving the accuracy of COVID-19 detection. Our method has demonstrated promising results on two public COVID-19 CXR datasets: a large dataset and an unbalanced dataset. It maintains high detection accuracy even when the number of annotated CXR training images is significantly reduced (e.g., utilizing only 10% of the original dataset). Furthermore, our method exhibits insensitivity to changes in hyperparameters, providing robust performance across different settings. In comparison to other state-of-the-art COVID-19 detection methods, our proposed approach outperforms them consistently. Therefore, our method has the potential to alleviate the workloads of healthcare providers and radiologists by providing a highly accurate COVID-19 detection solution using CXR images.

6.2 Method

An overview of the proposed method is depicted in Figure 6.1, illustrating the two essential phases involved. The first phase involves self-supervised learning-based fine-tuning, which aims to learn distinct representations from CXR images. In the second phase, batch knowledge ensembling-based fine-tuning is employed for accurate and automatic COVID-19 detection. Detailed explanations of these two phases can be found in subsections 6.2.1 and 6.2.2, respectively, providing a comprehensive understanding of the methodology.

6.2.1 Phase I: Self-Supervised Learning-based Pretraining

The first phase of our method is the self-supervised learning-based pretraining phase, which aims to learn distinctive representations from CXR images. This phase involves the utilization of an online network and a target network. The online network consists of an encoder E_θ, a projector G_θ, and a predictor P_θ. These components work together to extract meaningful features from the input CXR images. Similarly, the target network comprises an encoder E_ψ and a projector G_ψ. The target network acts as a reference for the online network during the training process. To generate pairs of views from an input CXR image x, two random transformations t_1 and t_2 are applied. These transformations are chosen randomly from a distribution T. The resulting pair of views are denoted as $v_1 = t_1(x_1)$ and $v_2 = t_2(x_2)$. To introduce diversity and increase robustness, various augmentation methods are employed during these transformations. These methods include cropping, resizing, flipping, color jittering, and Gaussian blur, among others. These augmentations enhance the ability of the model to learn meaningful and invariant representations from the CXR images.

During the self-supervised learning-based pretraining phase, the online network encoder E_θ and projector G_θ process the input view v_1. Similarly, the target network encoder E_ψ and projector G_ψ process the reference view v_2. To calculate the cross-view loss L_{CV}, we use v_2 as a

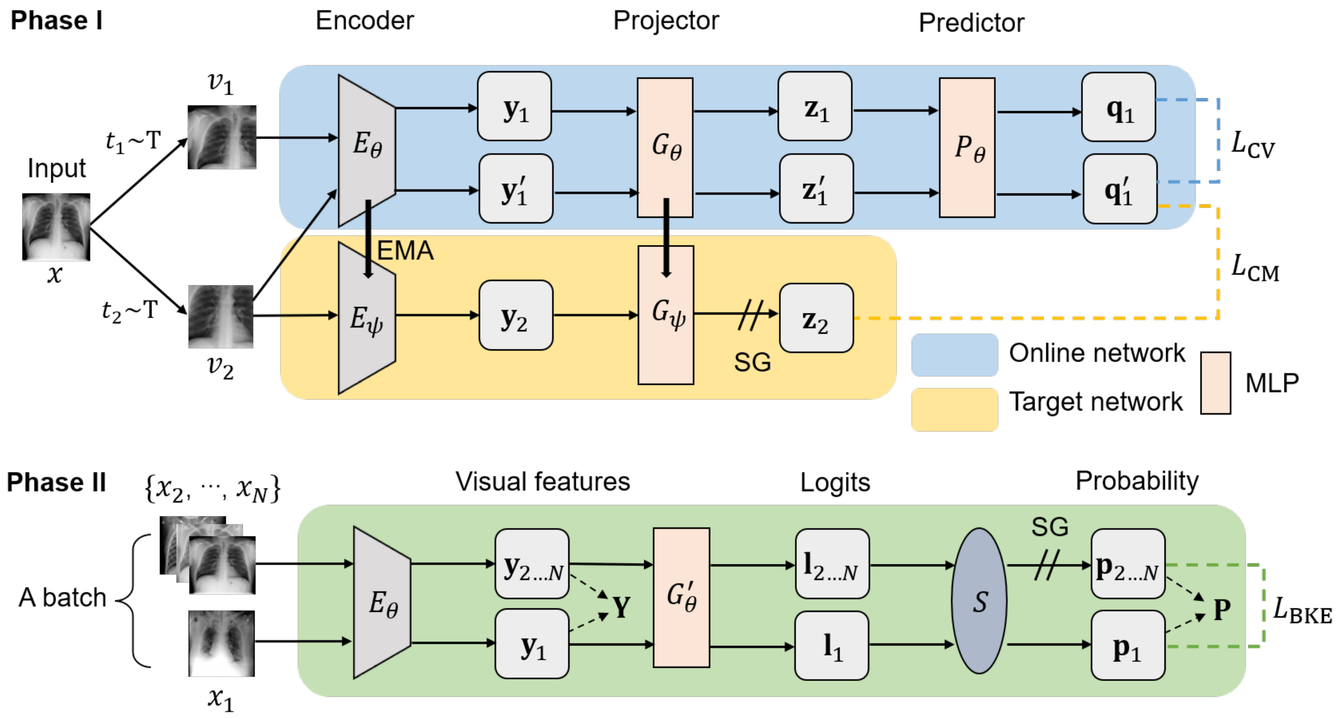


Figure 6.1: Overview of the proposed method.

reference and generate a transformed view by inputting it into the online network. This results in two sets of transformed features denoted as $\mathbf{q}_1$ and $\mathbf{q}_1'$, obtained using the predictor P_θ of the online network. The cross-view loss L_{CV} is computed as the squared Euclidean distance between the normalized features $\hat{\mathbf{q}}_1$ and $\hat{\mathbf{q}}_1'$ from the online network:

$$\begin{aligned} L_{\text{CV}} &= \|\hat{\mathbf{q}}_1 - \hat{\mathbf{q}}_1'\|_2^2 && (6.1) \\ &= 2 - 2 \cdot \frac{\left\langle \mathbf{q}_1, \mathbf{q}_1' \right\rangle}{\|\mathbf{q}_1\|_2 \cdot \|\mathbf{q}_1'\|_2}, && (6.2) \end{aligned}$$

Similarly, the cross-model loss L_{CM} is calculated as the squared Euclidean distance between the normalized features $\hat{\mathbf{q}}_1'$ from the online network and $\hat{\mathbf{z}}_2$ from the target network:

$$\begin{aligned} L_{\text{CM}} &= \|\hat{\mathbf{q}}_1' - \hat{\mathbf{z}}_2\|_2^2 && (6.3) \\ &= 2 - 2 \cdot \frac{\left\langle \mathbf{q}_1', \mathbf{z}_2 \right\rangle}{\|\mathbf{q}_1'\|_2 \cdot \|\mathbf{z}_2\|_2}, && (6.4) \end{aligned}$$

To update the weights of the online network (θ), the total loss $\mathcal{L}_{\theta,\psi}$ is computed as the sum of the cross-view loss and the cross-model loss:

$$\mathcal{L}_{\theta,\psi} = \mathcal{L}_{\text{CV}} + \mathcal{L}_{\text{CM}}. \tag{6.5}$$

The weights are then updated using an optimizer Opt and the gradients $\nabla\theta\mathcal{L}_{\theta,\psi}$ with a learning rate α:

$$\theta \leftarrow \text{Opt}(\theta, \nabla_\theta \mathcal{L}_{\theta,\psi}, \alpha). \tag{6.6}$$

Furthermore, the target network weights (ψ) are updated by applying an exponential moving average to the online network weights:

$$\psi \leftarrow \zeta\psi + (1 - \zeta)\theta. \tag{6.7}$$

To ensure stability during training, the target network's gradients are not backpropagated. Instead, a moving average is applied to update the target network weights. The degree of moving average is denoted by ζ. By averaging the target network weights with the online network weights, the target network slowly tracks the updates made by the online network over time.

In the subsequent fine-tuning phase, the learned parameters of the online network encoder E_θ are utilized. By leveraging the representations learned through self-supervised pretraining, the network can effectively extract relevant features from CXR images and achieve improved performance compared to training from scratch. The use of self-supervised learning in combination with fine-tuning enables the network to benefit from both the large-scale unlabeled data during pretraining and the task-specific labeled data during fine-tuning.

6.2.2 Phase II: Batch Knowledge Ensembling-based Fine-tuning

Next, we incorporate the batch knowledge ensembling-based fine-tuning phase of our method. This phase aims to enhance the classification performance of our model by leveraging the combined knowledge from images that exhibit similar visual features. The underlying assumption is that these images are likely to have similar predicted probabilities. Applying this concept to COVID-19 detection, we can leverage the similarities in visual features among different CXR images within a batch. By analyzing the collective knowledge of these images, we can improve the overall performance of our model in identifying COVID-19 cases. In this batch knowledge ensembling-based fine-tuning phase, we adopt a method that focuses on reducing redundancy and enhancing efficiency. By harnessing the power of ensemble learning, we combine the information from visually similar CXR images within a batch to refine the classification capabilities of our model. This approach allows us to leverage the shared characteristics and patterns among these images, leading to improved COVID-19 detection accuracy.

In the first step, we compute the visual feature similarity matrix, denoted as $\mathbf{Y} \in \mathbb{R}^{N \times N}$, using the encoded visual features $\mathbf{y}_1, ..., \mathbf{y}_N$ within a batch of N images. This matrix quantifies the similarities between different image pairs in terms of their visual features. To calculate $\mathbf{Y}$, we measure the cosine similarity between the normalized features $\hat{\mathbf{y}}_i = \mathbf{y}_i / ||\mathbf{y}_i||_2$ of image i and image j as follows:

$$\mathbf{Y}_{i,j} = (\hat{\mathbf{y}}_i^\top \hat{\mathbf{y}}_j). \tag{6.8}$$

To eliminate self-knowledge reinforcement, where an image's similarity with itself is considered, we modify the similarity matrix by applying the element-wise product with the complement of the identity matrix $\mathbf{I}$. This operation sets the diagonal entries of $\mathbf{Y}$ (representing self-similarity) to zero, effectively removing self-knowledge reinforcement. Therefore, the modified similarity matrix is given by $\mathbf{Y} = \mathbf{Y} \odot (1 - \mathbf{I})$. Next, we proceed with the normalization of the similarity matrix for visual features. We define the normalized similarity matrix as follows:

$$\hat{\mathbf{Y}}_{i,j} = \frac{\exp(\mathbf{Y}_{i,j})}{\sum_{j \neq i} \exp(\mathbf{Y}_{i,j})}, \forall i \in \{1, ..., N\}. \tag{6.9}$$

we apply a projector G'_θ and a softmax function S to the output logits $\{\mathbf{l}_1, ..., \mathbf{l}_N\}$ to obtain the predictive probabilities $\{\mathbf{p}_1, ..., \mathbf{p}_N\}$ as follows:

$$\mathbf{p}_{(k)} = \frac{\exp(\mathbf{l}_k/\tau)}{\sum_{i=1}^{K} \exp(\mathbf{l}_i/\tau)}, \tag{6.10}$$

where K denotes the total number of classes, and τ is a temperature hyperparameter that controls the softness of the probabilities. The probability matrix of a batch of CXR images is predicted as $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_N]^\top \in \mathbb{R}^{N \times K}$. To generate soft targets for batch knowledge ensembling, we combine the initial probability matrix $\mathbf{P}$ and the propagated probability matrix $\hat{\mathbf{Y}}\mathbf{P}$ using a weighted sum. This helps prevent the propagation of noisy predictions. The soft targets $\mathbf{Q}$ are calculated as follows:

$$\mathbf{Q} = \omega \hat{\mathbf{Y}} \mathbf{P} + (1 - \omega)\mathbf{P}. \tag{6.11}$$

Here, ω is a weight factor that determines the contribution of the propagated probability matrix compared to the initial probability matrix. To further enhance the soft targets for batch knowledge ensembling, we perform multiple propagation and ensemble iterations. The improved soft targets $\mathbf{Q}$ are generated as follows:

$$\mathbf{Q}_{(t)} = \omega \hat{\mathbf{Y}} \mathbf{Q}_{(t-1)} + (1 - \omega)\mathbf{P}, \tag{6.12}$$

$$= (\omega \hat{\mathbf{Y}})^t \mathbf{P} + (1 - \omega) \sum_{i=0}^{t-1} (\omega \hat{\mathbf{Y}})^i \mathbf{P}, \tag{6.13}$$

As the number of iterations approaches infinity, it can be observed that $\lim_{t \to \infty} (\omega \hat{\mathbf{Y}})^t = 0$ and $\lim_{t \to \infty} \sum_{i=0}^{t-1} (\omega \hat{\mathbf{Y}})^i = (\mathbf{I} - \omega \hat{\mathbf{Y}})^{-1}$. Based on this observation, we can approximate the inference formulation as follows:

$$\mathbf{Q} = (1 - \omega)(\mathbf{I} - \omega \hat{\mathbf{Y}})^{-1} \mathbf{P}. \tag{6.14}$$

Table 6.1: Details of the large COVID-19 CXR dataset [1].

Class	Full	Training set	Test set
COVID-19	3,616	2,893	723
Lung Opacity	6,012	4,810	1,202
Normal	10,192	8,154	2,038
Viral Pneumonia	1,345	1,076	269

Finally, we define the batch knowledge ensembling loss L_{BKE} as follows:

$$L_{\mathrm{BKE}} = L_{\mathrm{CE}} + \lambda \cdot \tau^2 \cdot D_{\mathrm{KL}}(\mathbf{Q}||\mathbf{P}), \tag{6.15}$$

The first component is the ordinary cross-entropy loss, denoted as L_{CE}. The second component involves the Kullback-Leibler divergence, denoted as $D_{\mathrm{KL}}(\mathbf{Q}||\mathbf{P})$, between the soft targets $\mathbf{Q}$ and the initial probability matrix $\mathbf{P}$. We introduce a balance hyperparameter λ to control the contribution of the Kullback-Leibler divergence term. Additionally, the temperature hyperparameter τ is squared to scale the Kullback-Leibler divergence term appropriately. During training, there is no backpropagation of gradients through the soft targets to ensure stable training.

In contrast to our previous method [75] which incorporated batch knowledge ensembling into the self-supervised learning phase, resulting in high memory usage and sensitivity to hyperparameters, we propose a modification where batch knowledge ensembling is integrated into the fine-tuning phase. This modification addresses the memory constraints associated with self-supervised learning while also improving the accuracy of COVID-19 detection. By leveraging the similarities in visual features among different CXR images, we enable the encoder E_θ to learn enhanced representations that can be utilized for COVID-19 detection. This approach capitalizes on the knowledge contained in visually similar images within a batch, allowing the model to extract more informative features and improve its performance in detecting COVID-19 cases.

6.3 Experiments

6.3.1 Dataset and Settings

In our study, we utilized two datasets: the large COVID-19 CXR dataset [1] and the COVID5K dataset [2]. The COVID-19 CXR dataset consists of a significant number of chest X-ray images and is well-balanced across four categories. The training and test sets were split in an 8:2 ratio.

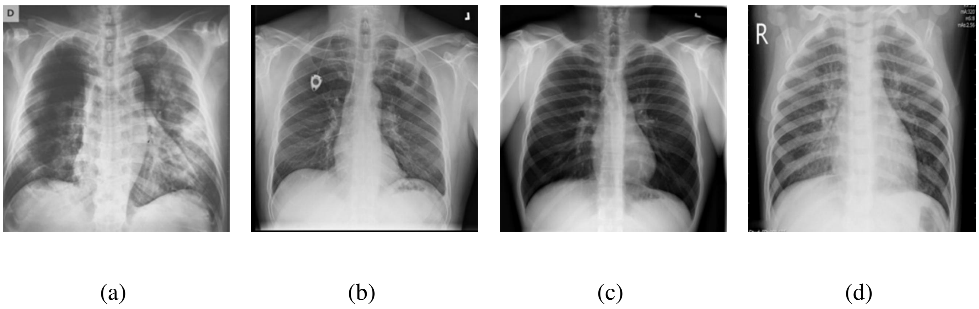


Figure 6.2: Examples of CXR images in the large COVID-19 CXR dataset [1]: (a) COVID-19, (b) Lung Opacity (c) Normal, and (d) Viral Pneumonia.

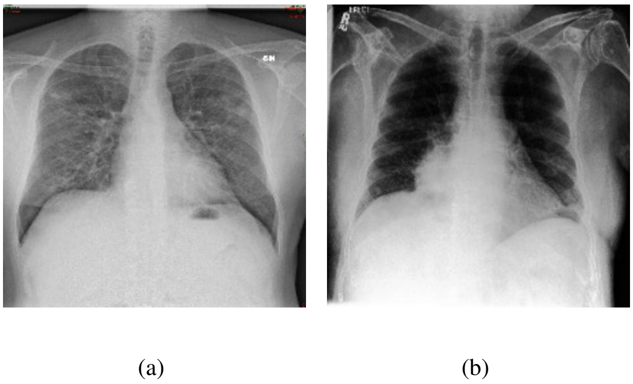


Figure 6.3: Examples of CXR images in the COVID5K dataset [2]: (a) COVID-19 , (b) Normal.

More details about this dataset can be found in Table 6.1.[1] On the other hand, the COVID5K dataset is an unbalanced dataset with a total of 5,520 images, of which only 520 images belong to the COVID-19 class. The dataset is divided into two classes, as shown in Table 6.2.[2] To provide visual examples, we have included Figures 6.2 and 6.3, which display some sample CXR images from both datasets. Each image in the datasets is grayscale and resized to a resolution of 224 pixels by 224 pixels. For evaluating the performance of our method, we employed several metrics including the area under the receiver operating characteristic curve (AUC), sensitivity (Sen), specificity (Spe), harmonic mean (HM) of sensitivity and specificity, and classification accuracy (Acc). In the context of COVID-19 detection, positive cases were determined based on Sen, Spe, HM, and AUC, while negative cases were assessed using the other metrics.

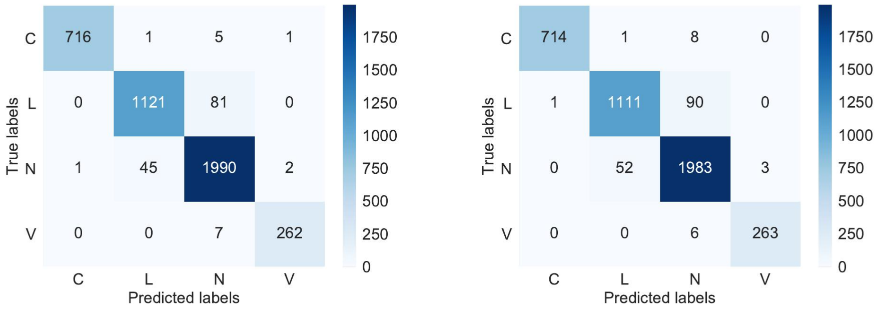
For our experiments, we employed either ResNet18 or ResNet50 as the encoder architec-

[1] https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database

[2] https://github.com/shervinmin/DeepCovid

Table 6.2: Details of the COVID5K dataset [2].

Class	Full	Training set	Test set
COVID-19	520	420	100
Normal	5,000	2,000	3,000



(a)

(b)

Figure 6.4: Best performance confusion matrix of our method. (a): ResNet50, (b): ResNet18.

ture [10]. The optimizer was stochastic gradient descent. The projectors and predictors utilized in our method were two-layer MLPs with the same structure as described in [41]. Our training process consisted of two phases: self-supervised learning and fine-tuning. We conducted 40 epochs of self-supervised learning followed by 30 epochs of fine-tuning on the datasets. To report our results, we calculated the average and variance of the performance metrics over the last 10 epochs of the fine-tuning phase. During the self-supervised learning-based pretraining phase, we set the batch size to 256. The generated view size, which refers to the size of the randomly generated views, was set to 112. Additionally, we used a moving average parameter ζ of 0.996. Data augmentation techniques such as cropping, resizing, flipping, and Gaussian blurring were employed to generate random views, enhancing the robustness and diversity of the learned representations. In the batch knowledge ensembling-based fine-tuning phase, we set the following hyperparameters: ω (weight factor) to 0.5, N (number of images in the batch) to 128, λ (balance hyperparameter) to 8.0, and τ (temperature hyperparameter) to 1.0. These settings were selected to ensure effective training and achieve the desired performance in COVID-19 detection, which was based on ablation studies.

We used several contrastive-based self-supervised learning methods as comparative methods, including BKE [75], Cross [71], BYOL [41], SimSiam [18], PIRL [38], and SimCLR [39]. To

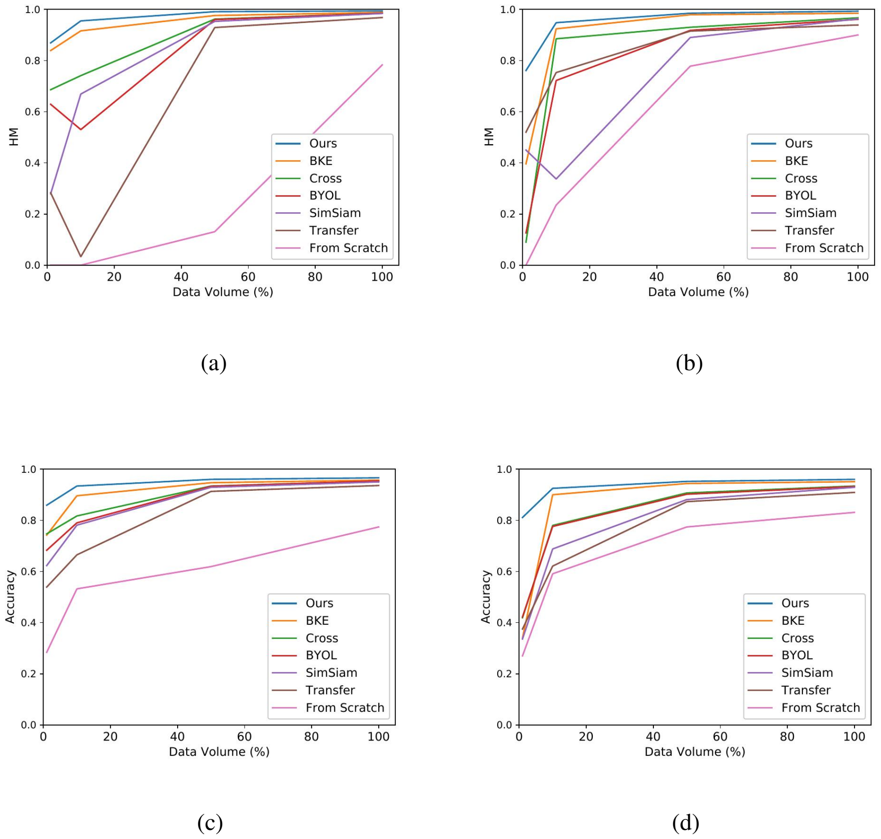
Table 6.3: Test accuracy on the large COVID-19 CXR Dataset.

Method	Structure	Sen	Spe	HM	AUC	Acc
Ours	ResNet50	**0.989±0.000**	**1.000±0.000**	**0.994±0.000**	**1.000±0.000**	**0.966±0.000**
BKE		0.980±0.004	0.997±0.001	0.988±0.002	0.999±0.000	0.957±0.001
Cross		0.972±0.003	0.997±0.001	0.985±0.001	0.999±0.000	0.953±0.001
BYOL		0.973±0.004	0.996±0.001	0.985±0.002	0.999±0.000	0.954±0.001
SimSiam		0.974±0.004	0.995±0.001	0.984±0.002	0.998±0.000	0.950±0.001
PIRL-Jigsaw		0.977±0.003	0.997±0.001	0.987±0.001	0.999±0.000	0.951±0.001
PIRL-Rotation		0.973±0.002	0.997±0.001	0.985±0.001	0.999±0.000	0.951±0.001
SimCLR		0.913±0.006	0.994±0.001	0.952±0.003	0.996±0.000	0.936±0.001
Transfer		0.944±0.004	0.994±0.001	0.968±0.002	0.997±0.000	0.936±0.001
From Scratch		0.665±0.013	0.954±0.003	0.783±0.008	0.935±0.001	0.774±0.002
Ours	ResNet18	**0.982±0.000**	**1.000±0.000**	**0.994±0.000**	**1.000±0.000**	**0.960±0.001**
BKE		0.972±0.004	0.998±0.000	0.985±0.002	1.000±0.000	0.951±0.001
Cross		0.944±0.003	0.990±0.001	0.967±0.001	0.996±0.000	0.934±0.002
BYOL		0.934±0.007	0.990±0.002	0.961±0.003	0.995±0.000	0.932±0.001
SimSiam		0.940±0.002	0.988±0.001	0.963±0.001	0.996±0.000	0.929±0.001
PIRL-Jigsaw		0.931±0.004	0.992±0.001	0.961±0.002	0.997±0.000	0.930±0.001
PIRL-Rotation		0.936±0.007	0.994±0.001	0.964±0.003	0.997±0.000	0.930±0.001
SimCLR		0.806±0.012	0.982±0.001	0.886±0.007	0.978±0.000	0.903±0.002
Transfer		0.900±0.008	0.981±0.003	0.939±0.003	0.993±0.000	0.909±0.001
From Scratch		0.849±0.010	0.958±0.004	0.900±0.004	0.974±0.000	0.831±0.001

Table 6.4: Test accuracy in different annotated data volumes when compared with vision transformer-based methods.

Method	Structure	1%	10%	50%	100%
Ours	ResNet50	**0.859**	**0.934**	**0.960**	**0.966**
Ours	ResNet18	0.811	0.925	0.952	0.960
RGMIM	ViT-Base	0.771	0.919	0.957	0.962
MAE	ViT-Base	0.754	0.903	0.948	0.956
Transfer	ViT-Base	0.689	0.893	0.940	0.953
From Scratch	ViT-Base	0.413	0.645	0.810	0.848

provide a comprehensive comparison, we also included two state-of-the-art methods [76, 77] based on masked image modeling using the vision transformer [13] architecture. In our experiments, both RGMIM and MAE utilized the ViT-Base model, which has shown promising results in various computer vision tasks. Additionally, we considered two baselines for comparison. The first baseline involved training the model from scratch, meaning that no pretraining was performed. The second baseline utilized transfer learning methods, leveraging pretrained models on large-scale datasets to initialize the encoder. To evaluate the COVID-19 detection accuracy using limited annotated data, we selected subsets of the training set consisting of 1%, 10%, and 50% of the original data, respectively. Importantly, the same selection ratio was applied to each category to ensure fair and consistent evaluation across different methods.



(a)

(b)

(c)

(d)

Figure 6.5: Test accuracy in different annotated data volumes: (a) HM of ResNet50, (b) HM of ResNet18, (c) Accuracy of ResNet50, and (d) Accuracy of ResNet18.

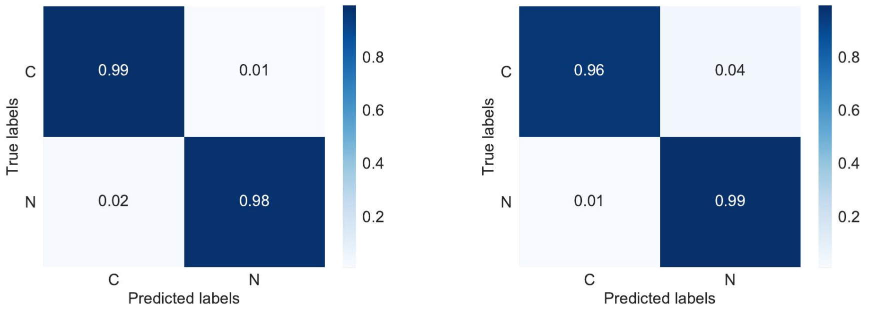
6.3.2 Test Accuracy on the Large COVID-19 CXR Dataset

The test accuracy of COVID-19 detection on the training data is provided in Table 6.3 for reference. When using the ResNet50 model on the entire training data, the transfer learning approach achieved HM, AUC, and Acc scores of 0.968, 0.997, and 0.936, respectively. Among the comparison methods, BKE [75] achieved the highest performance, with HM, AUC, and Acc scores of 0.988, 0.999, and 0.957, respectively. In contrast, the proposed method achieved superior results on the large COVID-19 CXR dataset, with HM, AUC, and Acc scores of 0.994, 1.000, and 0.966, respectively.

Figure 6.4 illustrates the confusion matrix of our method, demonstrating its excellent discrimination between patients with COVID-19 and normal patients. Additionally, our method achieves high accuracy in identifying COVID-19 cases and other types of pneumonia. The evaluation results under different settings indicate that our method shows promising performance in

Table 6.5: Test accuracy on the COVID5K dataset.

Method	Structure	Sen	Spe	HM	AUC
Ours	ResNet50	0.990±0.000	0.971±0.004	**0.980±0.002**	**0.997±0.000**
BKE		0.926±0.013	**0.989±0.001**	0.957±0.007	0.995±0.000
Cross		**0.999±0.005**	0.925±0.016	0.960±0.008	0.995±0.000
Transfer		0.961±0.010	0.908±0.013	0.934±0.005	0.984±0.001
From Scratch		0.818±0.026	0.916±0.016	0.864±0.009	0.930±0.002
Ours	ResNet18	0.958±0.004	**0.988±0.002**	**0.973±0.002**	**0.989±0.000**
BKE		0.939±0.003	0.973±0.002	0.955±0.002	**0.989±0.000**
Cross		**0.970±0.000**	0.946±0.007	0.958±0.004	0.987±0.000
Transfer		0.910±0.016	0.987±0.002	0.947±0.008	0.976±0.000
From Scratch		0.895±0.007	0.978±0.002	0.935±0.003	0.956±0.001



(a)

(b)

Figure 6.6: Best performance confusion matrix of our method. (a): ResNet50, (b): ResNet18.

COVID-19 detection on the large CXR dataset, surpassing other comparison methods and improving overall detection accuracy. It is worth noting that our method can run efficiently on a single NVIDIA Tesla P100 GPU with 16GB memory, whereas the previous method BKE [75] requires two GPUs. Furthermore, the training time of our method is approximately 97 minutes, while BKE takes around 124 minutes for training.

The COVID-19 detection results for different volumes of annotated data are presented in Table 6.4 and Fig. 6.5 for analysis. These results demonstrate the effectiveness of our method in improving COVID-19 detection even with a small amount of annotated data, such as 1% and 10% of the training set, which correspond to 169 and 1,693 images, respectively. Notably, our method achieved promising detection performance even with only 10% of the training set. Compared to the vision transformer-based methods RGMIM [76] and MAE [77], our method, which utilizes the traditional ResNet model, outperformed them, particularly when the amount of annotated data was significantly reduced. In real-world scenarios, the availability of annotated training data for COVID-19 may be limited due to various factors, such as differences in infection status, varying medical resources, and data-sharing policies across countries [78]. However, our method can still be applied effectively in such cases, enabling high-performance automatic COVID-19 detection.

6.3.3 Test Accuracy on the COVID5K Dataset

The test accuracy of COVID-19 detection on the training data and the corresponding confusion matrices of our method are presented in Table 6.5 and Fig. 6.6. The results reported are the average and variance of the last 10 fine-tuning epochs. In the case of using the ResNet50 architecture and utilizing all available training data, transfer learning achieved Sen, Spe, and HM scores of 0.961, 0.908, and 0.934, respectively. The best-performing comparison method, Cross [71], achieved Sen, Spe, and HM scores of 0.999, 0.925, and 0.960, respectively. On the other hand, our method achieved Sen, Spe, and HM scores of 0.990, 0.971, and 0.980, respectively, on the unbalanced COVID-19 CXR dataset. These results demonstrate that our method achieves promising detection performance even in the presence of imbalanced data, showcasing its robustness and applicability in real-world scenarios with extreme data situations. The presented results highlight the effectiveness and robustness of our method in COVID-19 detection, outperforming the comparison methods in terms of sensitivity, specificity, and the harmonic mean of these two metrics.

6.3.4 Exploring the Impact of Hyperparameters on Experimental Results

The evaluation results of different hyperparameters in the batch knowledge ensembling-based fine-tuning phase are presented in Tables 6.6 and 6.7. In our method, the hyperparameter ω controls the propagation of knowledge between the anchor CXR image and other images within the same batch during ensembling. By increasing the value of ω, the refined soft targets gain more information from other samples in the batch. To study the effects of the ensembling weight ω, we varied its value from 0.1 to 0.9. Interestingly, we observed that our method is relatively insensitive to the specific value of ω. Despite changes in ω, the overall performance of our method remained consistent. Furthermore, we found that as the value of ω increased, the harmonic mean (HM) scores decreased while the accuracy increased. This indicates a greater bias toward correctly detecting normalcy and other pneumonia cases, rather than focusing only on COVID-19 detection. These results provide insights into the influence of the ensembling weight ω in our method and its impact on the trade-off between overall accuracy and the ability to distinguish COVID-19 cases from other conditions.

First, we studied the influence of the batch size on our method by varying it from 32 to 512. Our findings indicate that our method achieved the best results when the batch size was set to 128. This suggests that a moderate batch size allows for effective utilization of the category information from different CXR images in the batch. Next, we examined the effect of the temperature parameter τ used to scale the predicted logits and soft targets. By increasing the value of τ from 2.0 to 16.0, we observed that our method remained relatively insensitive to temperature changes. However, the best results were obtained when τ was set to 8.0. This suggests that a moderate temperature value leads to a smoother probability distribution over classes, enhancing the model's performance. Furthermore, we explored the impact of the balance hyperparameter λ, which determines the trade-off between the cross-entropy loss and the batch knowledge ensembling loss. We varied λ from 0.5 to 4.0 and found that our method achieved the best results when λ was set to 1.0. This indicates that an equal weighting between the two loss components yields optimal performance. In summary, our investigations on the effects of hyperparameters demonstrate that our method is relatively robust to variations in the ensembling weight (ω), batch size, temperature (τ), and balance hyperparameter (λ). Nonetheless, specific values within certain ranges, such as a batch size of 128, τ of 8.0, and λ of 1.0, tend to yield the best performance for our method.

Robustness to hyperparameter changes is crucial in real-world clinical applications, where variations in shooting equipment and patient demographics are inevitable [79]. If a model is highly sensitive to hyperparameters, it would require extensive time and resources to fine-tune and optimize it for different settings, leading to inefficiencies and potential waste. The proposed method exhibits insensitivity to changes in hyperparameters, as evidenced by the evaluation results. This characteristic is promising for real-world clinical applications because it suggests that the model's performance remains stable and reliable across different regions, shooting equip-

Table 6.6: Evaluation results on the changes of the ensembling weight ω and the batch size N.

ω	HM	Acc	N	HM	Acc
0.1	0.996	0.964	32	0.993	0.961
0.3	0.996	0.967	64	0.994	0.963
0.5	0.994	0.966	128	0.994	0.966
0.7	0.994	0.967	256	0.993	0.962
0.9	0.993	0.967	512	0.992	0.964

Table 6.7: Evaluation results on the changes of the temperature τ and the weighting factor λ.

τ	HM	Acc	λ	HM	Acc
2	0.994	0.964	0.5	0.994	0.963
4	0.994	0.965	1	0.994	0.966
8	0.994	0.966	2	0.994	0.963
16	0.994	0.964	4	0.992	0.960

ment, and patient populations. By reducing the need for extensive hyperparameter adjustments, our method has the potential to save time, resources, and costs associated with model optimization, making it more practical and applicable in clinical settings.

6.3.5 Performance Comparison with Existing Methods

Table 6.8 provides a performance comparison between our method and existing approaches for COVID-19 detection from CXR images. While previous studies [80–87] have reported relatively high detection accuracy, it is important to note that these evaluations were typically conducted on small COVID-19 CXR image datasets with only two or three classes. Consequently, the applicability of these methods in real clinical situations may be limited. In contrast, our method was evaluated on a large COVID-19 CXR dataset containing four classes and a total of 3,616 COVID-19 images. Despite the increased complexity of the dataset, our method demonstrated promising detection performance. Notably, our approach utilizes the widely adopted and reliable ResNet50 model, which offers practical advantages in terms of its proven performance and widespread usage in various applications. By evaluating our method on a large and diverse dataset, we have provided evidence of its effectiveness and suitability for real-world clinical scenarios. The ability to perform well on a dataset with multiple classes and a substantial number of COVID-19 images underscores the robustness and practicality of our approach, further highlighting its potential for deployment in clinical settings.

Table 6.8: Performance comparison with the existing methods.

Method	Structure	Dataset	Accuracy
Narin et al. [80]	Inception-ResNetV2	COVID-19: 50, Normal: 50	Two-class: 0.980
Waheed et al. [81]	Auxiliary Classifier Generative Adversarial Network	COVID-19: 403, Normal: 721	Two-class: 0.950
Ozturk et al. [82]	DarkCovidNet	COVID-19: 127, Normal: 500	Two-class: 0.981
Zhang et al. [83]	ResNet34	COVID-19: 189, Normal: 235, Viral Pneumonia: 63	Three-class: 0.911
Togacar et al. [84]	Stacked models: MobileNetV2, SqueezeNet, SVM	COVID-19: 295, Normal: 65, Viral Pneumonia: 98	Three-class: 0.993
Gianchandani et al. [85]	Ensemble models: VGG16, ResNet152, DenseNet201	COVID-19: 423, Normal: 1,579, Viral Pneumonia: 1,485	Three-class: 0.962
Wang et al. [86]	COVID-Net	COVID-19: 358, Normal: 8,066, Viral Pneumonia: 5,538	Three-class: 0.933
Gour et al. et al. [87]	UA-ConvNet	COVID-19: 219, Normal: 1,341, Viral Pneumonia: 1,345	Three-class: 0.988
Ours	ResNet50	COVID-19: 3,616, Normal: 10,192, Viral Pneumonia: 1,345, Lung Opacity: 6,012	Four-class: 0.966

6.4 Conclusion

We have proposed a novel automatic COVID-19 detection method that leverages self-supervised learning and batch knowledge ensembling techniques using chest X-ray (CXR) images. By employing self-supervised learning-based pretraining, our method can extract meaningful representations from CXR images without relying on manually annotated labels. This enables the model to learn discriminative features that are crucial for detecting COVID-19. Additionally, our method incorporates batch knowledge ensembling-based fine-tuning, which takes advantage of the category information within a batch of images and exploits their visual feature similarities. This approach enhances the model's detection performance by leveraging the knowledge acquired from related images within the same batch. We evaluated our method on two public COVID-19 CXR datasets: a large dataset and an unbalanced dataset. In both cases, our method demonstrated promising COVID-19 detection performance. These results highlight the effectiveness of our approach in accurately identifying COVID-19 cases from CXR images.

Chapter 7

Efficient gastritis detection based on self-supervised representation learning

7.1 Introduction

The field of medical image analysis has seen significant advancements with the development of supervised learning techniques based on deep convolutional neural networks (DCNNs). However, annotating complex medical images often requires expert knowledge, limiting the applicability of these methods in real-world scenarios such as computer-aided diagnosis systems. To address this challenge, we propose a novel self-supervised learning method specifically designed for gastric X-ray images. Self-supervised learning allows models to learn discriminative representations without relying on explicit annotations. Our method leverages self-supervised learning to train a DCNN model on gastric X-ray images, enabling it to extract meaningful features and improve performance in gastritis detection. Through extensive experiments, we compared our proposed method with five state-of-the-art self-supervised learning methods and three previous approaches. The results demonstrated that our method outperformed all comparative methods, highlighting its superiority in learning discriminative representations from gastric X-ray images. Furthermore, the experimental results showcased the effectiveness of our self-supervised learning method in gastritis detection, even with limited annotated gastric X-ray images. This suggests the potential for clinical applications, where accurate diagnosis can be achieved using only a small number of annotated images.

7.2 Method

This section provides a comprehensive overview of the proposed approach. In subsection 7.2.1, we present a detailed description of the preprocessing steps applied to gastric X-ray images. Following that, in subsection 7.2.2, we showcase the proposed self-supervised learning technique. Lastly, we illustrate the process of fine-tuning and gastritis detection in subsection 7.2.3.

7.2.1 Gastric X-ray image preprocessing

The gastric X-ray images in our dataset have a resolution of 2,048 × 2,048 pixels at the patient level. However, due to the limited number of images available, we employ a patch-based approach to fully utilize the semantic information present in these images. Each patient-level image is divided into patches, and manual annotations are performed on these patches using the following three labels:

- $\mathcal{O}$: This label is assigned to patches located outside the stomach (outside patches),
- $\mathcal{N}$: Patches extracted from negative X-ray images inside the stomach (non-gastritis) are labeled as negative patches,
- $\mathcal{P}$: Patches extracted from positive X-ray images inside the stomach (gastritis) are labeled as positive patches.

7.2.2 Self-supervised learning

Our approach utilizes a teacher–student architecture for extracting informative features from gastric patches. An overview of the proposed method is presented in Figure 7.1. The teacher–student architecture consists of two networks that share the same structure. The weights of the teacher network are calculated as an exponential moving average of the weights of the student network, as described in [70]. The student network comprises three components: an encoder f_θ, a projector p_θ, and a predictor g_θ. These components work together to learn discriminative representations from the gastric patches. On the other hand, the teacher network consists of an encoder f_ψ and a projector g_ψ. By employing this teacher–student architecture, our method aims

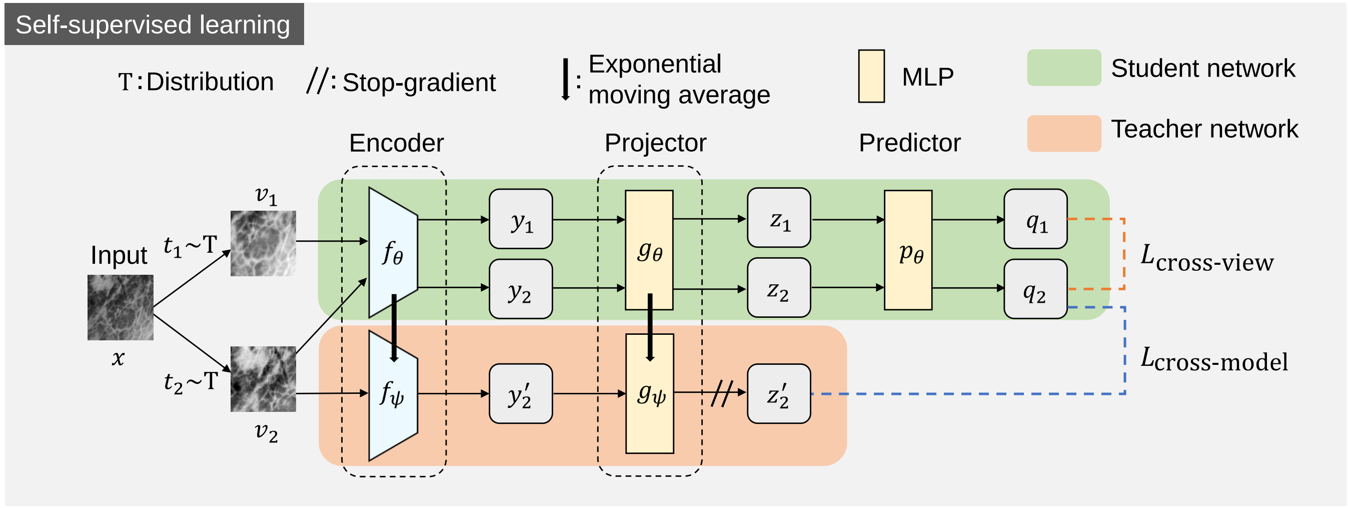


Figure 7.1: Overview of the proposed method.

to enhance the learning process and improve the quality of the extracted features from the gastric patches.

To generate two different views, denoted as v_1 and v_2, from an input patch x, we randomly sample two transformations, t_1 and t_2, from a distribution T. These transformations include standard data augmentation techniques such as cropping, resizing, flipping, and Gaussian blur, which are commonly used in self-supervised learning [69]. The view v_1 is processed by the encoder f_θ and projector g_θ of the student network. Similarly, the view v_2 undergoes processing by the encoder f_ψ and projector g_ψ of the teacher network, resulting in the final output z_2'. It's important to note that a copy of v_2 is created and input into the student network to calculate the final loss. Furthermore, the predictor p_θ, which is a multilayer perceptron (MLP) with a specific architecture, is used to transform the final outputs of the two views, producing q_1 and q_2 within the student network. The MLP architecture consists of a linear layer with an output size of 4,096, followed by a batch normalization layer, a ReLU activation function, and another linear layer with an output size of 256 [39].

The final step involves defining the cross-view and cross-model losses, which are used to guide the self-supervised learning process. Specifically, we perform self-supervised learning by reducing the distance between the representations of two views from the student network and reducing the distance between the representations of the same view from the teacher–student networks.

Cross-view loss. The cross-view loss is designed to compare the representations of two views obtained from the student network. It aims to penalize different predictions for views from positive pairs. The loss is defined by the following equation:

$$\begin{aligned} \mathcal{L}_{\text{cross-view}} &= ||\hat{q}_1 - \hat{q}_2||_2^2 \\ &= 2 - 2 \cdot \frac{\langle q_1, q_2 \rangle}{||q_1||_2 \cdot ||q_2||_2}, \end{aligned} \tag{7.1}$$

where $\hat{q}_1 = q_1/||q_1||_2$ and $\hat{q}_2 = q_2/||q_2||_2$ represent the normalized predictions of v_1 and v_2 from the student network, respectively.

Cross-model loss. The cross-model loss defined by the following equation compares the representations of the same view from the teacher–student networks, which penalizes different predictions and projections for the same view from different networks:

$$\begin{aligned}\mathcal{L}_{\text{cross-model}} &= \|\hat{q}_2 - \hat{z}'_2\|_2^2 \\ &= 2 - 2 \cdot \frac{\langle q_2, z'_2 \rangle}{\|q_2\|_2 \cdot \|z'_2\|_2},\end{aligned} \tag{7.2}$$

where $\hat{z}'_2 = z'_2/\|z'_2\|_2$ denotes the normalized projection of v_2 from the teacher network. To introduce asymmetry and prevent learning from collapsing, we incorporate the predictor component exclusively in the student network. This architectural asymmetry has been shown to enhance representation learning performance and prevent convergence issues [41]. The consistency between the views generated by the teacher-student networks plays a crucial role in learning discriminative representations from gastric patches. By aligning the representations of the same view obtained from both networks, we ensure that the student network benefits from the refined information provided by the teacher network. To update the weights of the student network (θ), we minimize a total loss that combines the cross-view and cross-model losses. The total loss $\mathcal{L}_{\theta,\psi}$ is defined as follows:

$$\mathcal{L}_{\theta,\psi} = \mathcal{L}_{\text{cross-view}} + \mathcal{L}_{\text{cross-model}}, \tag{7.3}$$

$$\theta \leftarrow \text{Opt}(\theta, \nabla_\theta \mathcal{L}_{\theta,\psi}, \alpha), \tag{7.4}$$

Opt and α denote an optimizer and the learning rate, respectively. To update the weights of the teacher network (ψ), we utilize an exponential moving average of the weights of the student network (θ). This updating process is performed after every iteration. The updating process is as follows:

$$\psi \leftarrow \tau\psi + (1 - \tau)\theta, \tag{7.5}$$

where τ denotes the degree of moving average. The weights of the teacher network are gradually updated to align with the weights of the student network. It is important to note that the teacher network's weights are not updated using backpropagation. This is because the stop-gradient operation plays a crucial role in preventing the collapse of self-supervised learning [18]. By fixing the weights of the teacher network, we ensure that the teacher–student architecture remains stable and effective for representation learning. Through self-supervised learning, the encoder component of the student network (f_θ) can effectively learn discriminative representations from the gastric patches. These learned representations can then be utilized for fine-tuning and gastritis detection tasks, leveraging the knowledge gained from the self-supervised learning process.

7.2.3 Fine-tuning and gastritis detection

After training the network using self-supervised learning, we proceed with fine-tuning the encoder component of the student network (f_θ) using a small set of annotated images. During the testing phase, we divide the patient-level gastric X-ray images into patches, following the approach described in subsection 7.2.1. Next, we load these divided patches into the fine-tuned deep convolutional neural network (DCNN) model and predict their labels. The predicted labels for the patches are used to estimate the numbers of positive patches, denoted as $\tilde{\mathcal{P}}$, and negative patches, denoted as $\tilde{\mathcal{N}}$. We do not consider the patches estimated as $\tilde{\mathcal{O}}$ since they correspond to regions outside the stomach and are not relevant for the final gastritis detection. Finally, we estimate the label of the patient-level gastric X-ray image based on the ratio of positive patches ($\tilde{\mathcal{P}}$) to the total number of patches ($\tilde{\mathcal{N}} + \tilde{\mathcal{P}}$), compared against a threshold σ:

$$y^{\text{test}} = \begin{cases} 1 & \text{if } \frac{\tilde{\mathcal{P}}}{\tilde{\mathcal{N}} + \tilde{\mathcal{P}}} \geq \sigma \\ 0 & \text{otherwise} \end{cases}, \tag{7.6}$$

Here, y^{test} represents the estimated label of the patient-level gastric X-ray image. If $y^{\text{test}} = 1$, the estimated label is positive, indicating the presence of gastritis. Conversely, if $y^{\text{test}} = 0$, the estimated label is negative, suggesting the absence of gastritis. The threshold σ can be adjusted based on specific experimental conditions and requirements.

7.3 Experiments

In this section, we present the experimental evaluation of the proposed method. We begin by describing the dataset used in the experiments in subsection 7.3.1. Subsequently, we discuss the experimental settings and present the results in subsections 7.3.2 and 7.3.3, respectively.

7.3.1 Dataset

For our experiments, we utilized a dataset consisting of gastric X-ray images from 815 patients (240 positive and 575 negative cases). The resolution of these gastric X-ray images is set at 2,048 × 2,048 pixels. Each image in the dataset is assigned a ground truth label indicating whether it is positive (indicating the presence of gastritis) or negative (indicating the absence of gastritis), based on the diagnostic results obtained from X-ray inspection and endoscopic examination. Out of the total dataset, we selected 200 patient images (100 positive and 100 negative) as the training set, while the remaining images were used for testing. As mentioned in section 2.1, we divided the gastric X-ray images into patches. The patch size was set to 299 pixels, and the sliding interval was set to 50 pixels, as described in our previous study [88]. For the training data, the patches were annotated as $\mathcal{O}$ (outside the stomach), $\mathcal{N}$ (negative), or $\mathcal{P}$ (positive) by a radiological technologist. If the area inside the stomach in a patch was less than 1%, it was annotated as $\mathcal{O}$. Conversely, if the area inside the stomach was greater than 85%, the patch was annotated as either $\mathcal{N}$ or $\mathcal{P}$. Patches that did not meet these criteria were discarded. As a result, the numbers of obtained $\mathcal{O}$, $\mathcal{N}$, and $\mathcal{P}$ patches were 48,385, 42,785, and 45,127, respectively. These patches formed the training data for the proposed method.

The dataset consisting of 200 patients' patches was used as the training set for the self-supervised learning process, without utilizing their label information. To further evaluate the performance of the model, we divided these 200 patients' patches into two subsets: 120 patients' patches for training and 80 patients' patches for validation during the fine-tuning process. The training set was evenly split between positive and negative patches.

Additionally, we randomly selected subsets of 10, 20, 30, and 40 patients' patches, equally divided between positive and negative patches, from the training set for the fine-tuning process. This allowed us to investigate the impact of different amounts of annotated data on the model's performance.

A detailed illustration of the partitioned data used in this study is provided in Figure 7.2.

7.3.2 Implementation

The size of the generated views was set to 128 in our experiments. We utilized the ResNet50 network as the encoder for both f_θ and f_ψ, with an output feature dimension of 2,048, obtained

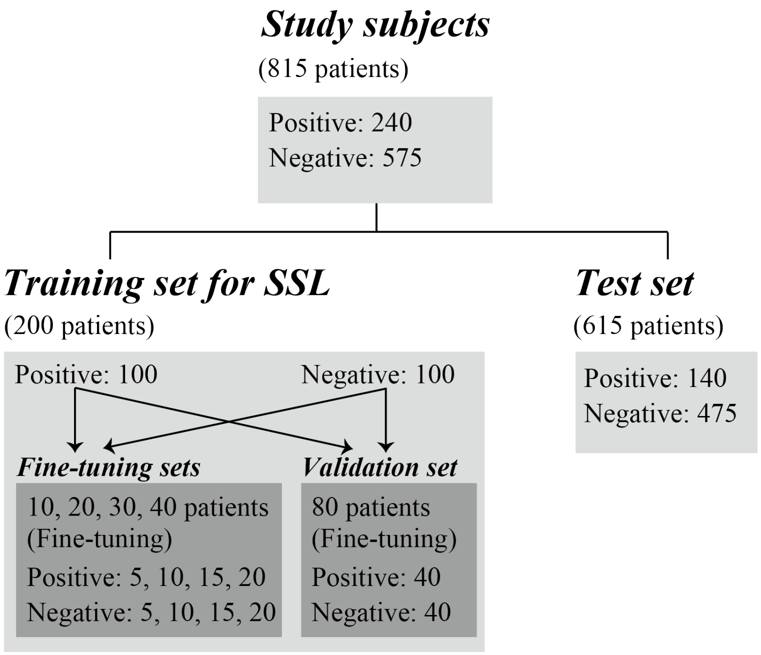


Figure 7.2: Details of the partitioned datasets used in the present study. SSL denotes self-supervised learning process.

Table 7.1: Hyperparameters of the proposed method.

Parameter	Value
Epoch	80
Batch size	256
Learning rate (α)	0.03
momentum	0.9
weight decay	0.0004
moving average (τ)	0.996
mlp hidden size	4096
projection size	256
View size	128
Threshold (σ)	0.5

from the final average pooling layer. For optimization, we employed the SGD optimizer with a learning rate of α = 0.03, momentum of 0.9, and weight decay of 0.0004. In the self-supervised learning process, we trained the model for 80 epochs. The hyperparameters of our proposed method are summarized in Table 7.1.

During the fine-tuning process, we initialized the weights of the student network's encoder

(f_θ) with the trained weights obtained from self-supervised learning. We hypothesize that the effectiveness of self-supervised learning positively impacts the performance of gastritis detection. To validate the effectiveness of our method (referred to as ”Ours”), we compared it against the following state-of-the-art self-supervised learning methods: SimSiam [18], BYOL [41], PIRL-Jigsaw [38], PIRL-Rotation [38], and SimCLR [39]. Additionally, we included our previous methods as baselines, including semi-supervised learning based on tri-training [89], transfer learning [90] using ImageNet pre-trained weights, and training from scratch.

In our experiments, we conducted comparisons between our proposed method and state-of-the-art self-supervised learning methods using all four fine-tuning sets. We also compared our method with our previous methods using the full training set. The performance of gastritis detection was evaluated on the test set, which consisted of 615 patients' data. For SimSiam and BYOL, we strictly followed the same settings as our method since they do not require negative sample pairs. The settings for PIRL and SimCLR, which do require negative sample pairs, were guided by previous works [38,39], with the exception of the number of negative sample pairs due to our computing resource limitations. To ensure a fair comparison, we directly used the results reported in [89] for semi-supervised learning based on tri-training. Furthermore, we maintained the same settings for the fine-tuning process across all the experiments. Please note that specific details of the settings and results for each method can be found in the corresponding sections of the paper.

In the test phase, we set the threshold σ experimentally to 0.5 to achieve a high gastric detection performance. To evaluate the performance, we used sensitivity (Sen), specificity (Spe), and the harmonic mean (HM) as evaluation metrics:

$$\mathrm{Sen} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{7.7}$$

$$\mathrm{Spe} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}, \tag{7.8}$$

$$\mathrm{HM} = \frac{2 \times \mathrm{Sen} \times \mathrm{Spe}}{\mathrm{Sen} + \mathrm{Spe}}, \tag{7.9}$$

where TP, TN, FP, and FN represent the number of true positive, true negative, false positive, and false negative, respectively. A higher sensitivity means maintaining a high ability to correctly identify positive cases, while decreasing specificity may lead to a higher number of false positives. Calculating the harmonic mean (HM) between sensitivity and specificity holds equal value as it balances the trade-off between these two metrics, providing a comprehensive evaluation of the detection performance.

7.3.3 Results

The experimental results are presented in Tables 7.2 and 7.3. Table 7.2 displays the patient-level gastritis detection results after fine-tuning with annotated data from 10, 20, 30, and 40 patients, allowing for a comparison with state-of-the-art self-supervised learning methods. According to Table 7.2, our method demonstrates superior performance in gastritis detection. For instance, our method achieves HM scores of sensitivity and specificity of 0.875, 0.911, 0.915, and 0.931 after fine-tuning with annotated data from 10, 20, 30, and 40 patients, respectively. The average HM scores of our method across the four randomly selected training sets are 0.034, 0.021, 0.153, 0.293, and 0.351, respectively, outperforming other state-of-the-art methods. Additionally, the most commonly used pretrained model of ImageNet achieves an HM score of 0.870 with annotations from 40 patients. In contrast, our method achieves an HM score of 0.875 with only 10 patients' annotations. These experimental results indicate that our method achieves high gastritis detection performance with only a few annotations, substantially reducing the need for manual labeling. Table 7.3 presents the patient-level gastritis detection results after fine-tuning with varying numbers of annotated patients' data, enabling a comparison with our previous methods. From Table 7.3, it is evident that our method not only significantly outperforms previous methods with a small amount of annotated data but also achieves excellent detection performance as the number of annotated data increases.

7.4 Conclusion

In this study, we introduced a novel self-supervised learning method that utilizes a teacher-student architecture for gastritis detection using gastric X-ray images. Our method incorporates cross-view and cross-model losses to enable explicit self-supervised learning and to learn discriminative representations from gastric X-ray images. The experimental results demonstrate the

	10 patients			20 patients			30 patients			40 patients		
Method	Sen	Spe	HM	Sen	Spe	HM	Sen	Spe	HM	Sen	Spe	HM
Ours	0.957	0.806	**0.875**	0.964	0.863	0.911	0.936	0.895	**0.915**	0.964	0.901	**0.931**
SimSiam [18]	0.950	0.739	0.831	0.964	0.771	0.857	0.921	0.863	0.891	0.907	0.928	0.918
BYOL [41]	0.964	0.758	0.849	0.907	0.920	**0.913**	0.807	0.956	0.875	0.964	0.861	0.910
PIRL-Jigsaw [38]	0.721	0.806	0.761	0.943	0.507	0.659	0.879	0.743	0.805	0.864	0.737	0.795
PIRL-Rotation [38]	0.607	0.735	0.665	0.929	0.316	0.472	0.879	0.440	0.586	0.886	0.632	0.738
SimCLR [39]	0.707	0.274	0.395	0.521	0.617	0.565	0.407	0.762	0.531	0.479	0.861	0.615
Transfer [90]	0.236	1.000	0.382	0.850	0.686	0.759	0.279	0.989	0.435	0.864	0.876	0.870
Baseline	0.771	0.225	0.348	0.693	0.364	0.477	0.457	0.734	0.563	0.579	0.726	0.644

Table 7.2: Comparison with the state-of-the-art self-supervised learning methods.

Table 7.3: Comparison with our previous methods.

Method	10 patients	20 patients	30 patients	40 patients	200 patients
Ours	**0.875**	**0.911**	**0.915**	**0.931**	**0.954**
Tri-training [89]	0.860	0.870	-	-	0.922
Transfer [90]	0.382	0.759	0.435	0.870	**0.954**
Baseline	0.348	0.477	0.563	0.644	0.876

effectiveness of our proposed method. We achieved high patient-level gastritis detection performance even with only a small number of annotations. This implies that our method has the potential to significantly reduce the reliance on manual annotations while still achieving accurate detection results.

Chapter 8

Conclusions

As conclusions of this thesis, this chapter reviews the overview of the proposition and shows future directions.

8.1 Overview of the Proposition in this Thesis

This section provides an overview of the proposition presented in this thesis, building upon the background and previous sections.

The proposition of this thesis is to construct new datasets more efficiently and to enhance the learning capabilities of models when facing extremely limited data or labels. To achieve this goal, the thesis proposes a novel data-efficient learning method consisting of the following three stages. The first stage involves assessing the complexity of datasets by analyzing their characteristics and properties. Understanding the complexities of a dataset allows researchers to make well-informed choices regarding model architecture, training strategies, and data augmentation techniques that are appropriate for that particular dataset. This stage plays a crucial role in optimizing the learning process and achieving superior performance with limited data. Building upon the dataset complexity assessment, the second stage introduces the concept of dataset distillation. Dataset distillation leverages knowledge from a larger, labeled dataset to distill it into a smaller, more compact dataset. The distilled dataset retains the most relevant information that is essential for the target task. This stage can enhance data processing efficiency and avoid overfitting or noise from the large dataset. Lastly, the third stage explores self-supervised learning as a data-efficient learning method. Self-supervised learning involves training models to solve pretext tasks using unlabeled data, with labels generated automatically or through heuristics. The learned representations from these pretext tasks can then be transferred to the target task, effectively utilizing the large amounts of unlabeled data to improve performance. This stage can reduce reliance on labeled data while still achieving competitive results. With the incorporation of the three stages, the newly proposed data-efficient learning method can effectively address the existing challenges. Below, the overview of each chapter of this thesis is reviewed.

In Chapter 2, related works of data-efficient learning are presented and problems to be solved are clarified. In Chapter 3, a dataset complexity assessment method based on spectral clustering was presented. Chapter 4 proposed a method of generation of compressed gastric images based on soft-label dataset distillation for efficient anonymous medical data sharing. Chapters 5 and 6 presented self-supervised learning methods for COVID-19 detection from chest X-ray images. In Chapter 7, a self-supervised learning method for learning discriminative representations from gastric X-ray images was presented.

The contributions of this thesis can be summarized as follows:

- The thesis introduces a new data-efficient learning method that encompasses three stages: dataset complexity assessment, dataset distillation, and self-supervised learning. The proposed method aims to construct new datasets with improved efficiency and enhance model learning capabilities, particularly when dealing with severely limited data or labels.

- The effectiveness of the proposed method is evaluated on both natural image datasets and medical image datasets. The proposed method extends the existing knowledge and techniques in data-efficient learning and provides valuable insights for researchers and practitioners working in this area.

8.2 Future Directions

This section describes the future directions of this study. Future work can focus on refining and optimizing the proposed data-efficient learning methods. This includes fine-tuning the dataset complexity assessment approach to consider additional factors that impact dataset complexity, such as class imbalance or label noise. Similarly, the dataset distillation method can be enhanced by investigating different strategies for selecting representative samples and evaluating the trade-off between the size of the distilled dataset and model performance. Additionally, the self-supervised learning approach can be further explored to develop more effective pretext tasks that capture relevant information for the target task.

While the thesis focuses on the application of data-efficient learning methods in the medical domain, these methods can be extended to other domains as well. Future research can explore the effectiveness of the proposed methods in fields such as finance, natural language processing, robotics, and more. Each domain presents unique challenges and characteristics, and adapting the data-efficient learning methods to these domains can provide valuable insights and advancements.

Investigating the integration of multiple data-efficient learning methods can be an interesting direction for future research. Combining the strengths of different approaches, such as dataset complexity assessment, dataset distillation, and self-supervised learning, can potentially lead to even more powerful and robust data-efficient learning frameworks. Developing effective strategies for integrating these methods and understanding their synergistic effects will be crucial for further enhancing data efficiency.

As data-efficient learning methods continue to evolve and find applications in various domains, it is important to explore the ethical implications associated with these techniques. Future research can delve into the ethical considerations related to data privacy, bias, fairness, and transparency in data-efficient learning. Developing frameworks and guidelines to ensure responsible and ethical use of limited data resources will be crucial for the widespread adoption of these methods.

Acknowledgements

First, I would like to sincerely thank my supervisors, Prof. Miki Haseyama and Prof. Takahiro Ogawa. This thesis would not have been possible without the invaluable guidance and encouragement they have given me over the four years I spent at the Graduate School of Information Science and Technology, Hokkaido University.

I would also like to thank Specially Appointed Prof. Kenji Araki, Specially Appointed Prof. Yuji Sakamoto, and Prof. Yoshinori Dobashi for providing insightful comments and suggestions about the research I performed at the Graduate School of Information Science and Technology, Hokkaido University.

Furthermore, I would like to sincerely thank Specially Appointed Assistant Prof. Ren Togo, Specially Appointed Assistant Prof. Keisuke Maeda, and Assistant Prof. Naoki Saito for their constant encouragement and advice about research and academic life.

Finally, I would like to sincerely thank everyone at the Laboratory of Media Dynamics, Graduate School of Information Science and Technology, Hokkaido University, and my family for their invaluable support and assistance.

References

[1] T. Rahman, A. Khandakar, Y. Qiblawey, A. Tahir, S. Kiranyaz, S. B. A. Kashem, M. T. Islam, S. Al Maadeed, S. M. Zughaier, M. S. Khan, et al., "Exploring the effect of image enhancement techniques on covid-19 detection using chest x-ray images," *Computers in Biology and Medicine*, vol. 132, pp. 104319, 2021.

[2] S. Minaee, R. Kafieh, M. Sonka, S. Yazdani, and G. J. Soufi, "Deep-covid: Predicting covid-19 from chest x-ray images using deep transfer learning," *Medical Image Analysis*, vol. 65, pp. 101794, 2020.

[3] F. Branchaud-Charron, A. Achkar, and P.-M. Jodoin, "Spectral metric for dataset complexity assessment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3215–3224.

[4] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[5] M. Yutaka, L. Yann, S. Maneesh, P. Doina, S. David, S. Masashi, U. Eiji, and M. Jun, "Deep learning, reinforcement learning, and world models," *Neural Networks*, vol. 152, pp. 267–275, 2022.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[7] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Computational Intelligence Magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[8] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, et al., "Deep speech 2: End-to-end speech recognition

in english and mandarin," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2016, pp. 173–182.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2012, pp. 1097–1105.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

[12] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2020, pp. 12449–12460.

[13] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[14] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., "Learning transferable visual models from natural language supervision," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2021, pp. 8748–8763.

[15] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 10684–10695.

[16] O. AI, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[17] H.-N. Dai, R. C.-W. Wong, H. Wang, Z. Zheng, and A. V. Vasilakos, "Big data analytics for large-scale wireless networks: Challenges and opportunities," *ACM Computing Surveys*, vol. 52, no. 5, pp. 1–36, 2019.

[18] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[19] G. Nguyen, S. Dlugolinsky, M. Bobák, V. Tran, Á. López García, I. Heredia, P. Malík, and L. Hluchỳ, "Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey," *Artificial Intelligence Review*, vol. 52, pp. 77–124, 2019.

[20] L. Zhou, S. Pan, J. Wang, and A. V. Vasilakos, "Machine learning on big data: Opportunities and challenges," *Neurocomputing*, vol. 237, pp. 350–361, 2017.

[21] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[22] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A survey of deep active learning," *ACM Computing Surveys*, vol. 54, no. 9, pp. 1–40, 2021.

[23] X. Yang, Z. Song, I. King, and Z. Xu, "A survey on deep semi-supervised learning," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[24] D. Zhang, J. Han, G. Cheng, and M.-H. Yang, "Weakly supervised object localization and detection: A survey," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5866–5885, 2021.

[25] T. K. Ho and M. Basu, "Complexity measures of supervised classification problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 289–300, 2002.

[26] L. Wang, Y. Zhang, and J. Feng, "On the euclidean distance of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1334–1339, 2005.

[27] R. Baumgartner and R. L. Somorjai, "Data complexity assessment in undersampled classification of high-dimensional biomedical data," *Pattern Recognition Letters*, vol. 27, no. 12, pp. 1383–1389, 2006.

[28] A. Orriols-Puig, N. Macia, and T. K. Ho, "Documentation for the data complexity library in c++," *Universitat Ramon Llull, La Salle*, vol. 196, pp. 1–40, 2010.

[29] L. P. Garcia, A. C. d. Carvalho, and A. C. Lorena, "Effect of label noise in the complexity of classification problems," *Neurocomputing*, vol. 160, pp. 108–119, 2015.

[30] S. Har-Peled and S. Mazumdar, "On coresets for k-means and k-median clustering," in *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, 2004, pp. 291–300.

[31] D. Feldman, M. Schmidt, and C. Sohler, "Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering," in *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013, p. 1434–1453.

[32] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2001–2010.

[33] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari, "End-to-end incremental learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 233–248.

[34] K. Killamsetty, D. Sivasubramanian, G. Ramakrishnan, and R. Iyer, "Glister: Generalization based data subset selection for efficient and robust learning," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021, vol. 35, pp. 8110–8118.

[35] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros, "Dataset distillation," *arXiv:1811.10959*, 2018.

[36] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 69–84.

[37] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[38] I. Misra and L. v. d. Maaten, “Self-supervised learning of pretext-invariant representations,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6707–6717.

[39] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[40] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum contrast for unsupervised visual representation learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9729–9738.

[41] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al., “Bootstrap your own latent-a new approach to self-supervised learning,” in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[42] W. Wang, Y. Huang, Y. Wang, and L. Wang, “Generalized autoencoder: A neural network framework for dimensionality reduction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Workshop*, 2014, pp. 490–497.

[43] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[44] S. Wold, K. Esbensen, and P. Geladi, “Principal component analysis,” *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[45] E. Nowakowska, J. Koronacki, and S. Lipovetsky, “Tractable measure of component overlap for gaussian mixture models,” *arXiv preprint arXiv:1407.7172*, 2014.

[46] T. Jebara, R. Kondor, and A. Howard, “Probability product kernels,” *Journal of Machine Learning Research*, vol. 5, pp. 819–844, 2004.

[47] K. Binder, D. Heermann, L. Roelofs, A. J. Mallinckrodt, and S. McKay, “Monte carlo simulation in statistical physics,” *Computers in Physics*, vol. 7, no. 2, pp. 156–157, 1993.

[48] U. Von Luxburg, “A tutorial on spectral clustering,” *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[49] B. Mohar, “Some applications of laplace eigenvalues of graphs,” in *Graph Symmetry*, pp. 225–275. Springer, 1997.

[50] E. W. Beals, “Bray-curtis ordination: an effective strategy for analysis of multivariate ecological data,” *Advances in Ecological Research*, vol. 14, pp. 1–55, 1984.

[51] Y. LeCun, C. Cortes, and C. Burges, “Mnist handwritten digit database,” [Online]. Available: `http://yann.lecun.com/exdb/mnist/`, 2010.

[52] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Workshop*, 2011.

[53] A. Krizhevsky, G. Hinton, et al., “Learning multiple layers of features from tiny images,” 2009.

[54] Y. Bulatov, “Notmnist dataset,” [Online]. Available: `http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html`, 2011.

[55] A. Coates, A. Ng, and H. Lee, “An analysis of single-layer networks in unsupervised feature learning,” in *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011, pp. 215–223.

[56] L. Yang, P. Luo, C. Change Loy, and X. Tang, “A large-scale car dataset for fine-grained categorization and verification,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3973–3981.

[57] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1251–1258.

[58] M. Tan and Q. V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2019, pp. 6105–6114.

[59] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10687–10698.

[60] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Soft-label anonymous gastric x-ray image distillation," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 305–309.

[61] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS) Workshops*, 2014.

[62] G. Li, B. Zhao, and T. Wang, "Awesome-dataset-distillation," `https://github.com/Guang000/Awesome-Dataset-Distillation`, 2022.

[63] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[64] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[65] X. Yang, X. He, Y. Liang, Y. Yang, S. Zhang, and P. Xie, "Transfer learning or self-supervised learning? a tale of two pretraining paradigms," *arXiv preprint arXiv:2007.04234*, 2020.

[66] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255.

[67] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[68] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Proceedings of the Advances in Neural Information*

Processing Systems (NeurIPS), 2020.

[69] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola, "What makes for good views for contrastive learning," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[70] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 1195–1204.

[71] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Self-supervised learning for gastritis detection with gastric x-ray images," *International Journal of Computer Assisted Radiology and Surgery*, pp. 1–8, 2023.

[72] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Cross-view self-supervised learning via momentum statistics in batch normalization," in *Proceedings of the IEEE International Conference on Consumer Electronics – Taiwan (ICCE-TW)*, 2021, pp. 1–2.

[73] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 839–847.

[74] L. Brunese, F. Mercaldo, A. Reginelli, and A. Santone, "Explainable deep learning for pulmonary disease and coronavirus covid-19 detection from x-rays," *Computer Methods and Programs in Biomedicine*, vol. 196, pp. 105608, 2020.

[75] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Self-knowledge distillation based self-supervised learning for covid-19 detection from chest x-ray images," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 1371–1375.

[76] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Rgmim: Region-guided masked image modeling for covid-19 detection," *arXiv preprint arXiv:2211.00313*, 2022.

[77] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF International Conference on Computer*

Vision (CVPR), 2022, pp. 16000–16009.

[78] Y. N. Alhalaseh, H. A. Elshabrawy, M. Erashdi, M. Shahait, A. M. Abu-Humdan, and M. Al-Hussaini, "Allocation of the " already " limited medical resources amid the covid-19 pandemic, an iterative ethical encounter including suggested solutions from a real life encounter," *Frontiers in Medicine*, vol. 7, pp. 1076, 2021.

[79] D. Zimmerer, F. Isensee, J. Petersen, S. Kohl, and K. Maier-Hein, "Unsupervised anomaly localization using variational auto-encoders," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2019, pp. 289–297.

[80] A. Narin, C. Kaya, and Z. Pamuk, "Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks," *Pattern Analysis and Applications*, vol. 24, pp. 1207–1220, 2021.

[81] A. Waheed, M. Goyal, D. Gupta, A. Khanna, F. Al-Turjman, and P. R. Pinheiro, "Covidgan: data augmentation using auxiliary classifier gan for improved covid-19 detection," *IEEE Access*, vol. 8, pp. 91916–91923, 2020.

[82] T. Ozturk, M. Talo, E. A. Yildirim, U. B. Baloglu, O. Yildirim, and U. R. Acharya, "Automated detection of covid-19 cases using deep neural networks with x-ray images," *Computers in Biology and Medicine*, vol. 121, pp. 103792, 2020.

[83] R. Zhang, Z. Guo, Y. Sun, Q. Lu, Z. Xu, Z. Yao, M. Duan, S. Liu, Y. Ren, L. Huang, et al., "Covid19xraynet: a two-step transfer learning model for the covid-19 detecting problem based on a limited number of chest x-ray images," *Interdisciplinary Sciences: Computational Life Sciences*, vol. 12, pp. 555–565, 2020.

[84] M. Toğaçar, B. Ergen, and Z. Cömert, "Covid-19 detection using deep learning models to exploit social mimic optimization and structured chest x-ray images using fuzzy color and stacking approaches," *Computers in Biology and Medicine*, vol. 121, pp. 103805, 2020.

[85] N. Gianchandani, A. Jaiswal, D. Singh, V. Kumar, and M. Kaur, "Rapid covid-19 diagnosis using ensemble deep transfer learning models from chest radiographic images," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–13, 2020.

[86] L. Wang, Z. Q. Lin, and A. Wong, "Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images," *Scientific Reports*, vol. 10, no. 1, pp. 1–12, 2020.

[87] M. Gour and S. Jain, "Uncertainty-aware convolutional neural network for covid-19 x-ray images classification," *Computers in Biology and Medicine*, vol. 140, pp. 105047, 2022.

[88] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Complexity evaluation of medical image data for classification problem based on spectral clustering," in *Proceedings of the IEEE Global Conference on Consumer Electronics (GCCE)*, 2020, pp. 667–669.

[89] Z. Li, R. Togo, T. Ogawa, and M. Haseyama, "Chronic gastritis classification using gastric x-ray images with a semi-supervised learning method based on tri-training," *Medical & Biological Engineering & Computing*, vol. 58, no. 6, pp. 1239–1250, 2020.

[90] M. Kanai, R. Togo, T. Ogawa, and M. Haseyama, "Gastritis detection from gastric x-ray images via fine-tuning of patch-based deep convolutional neural network," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 1371–1375.

Achievements of the Author

(A) **Journal**

[**A-1**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “Dataset complexity assessment based on cumulative maximum scaled area under Laplacian spectrum,” *Multimedia Tools and Applications* (IF: 3.6), vol. 81, no. 22, pp. 32287–32303, 2022.

[**A-2**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “Compressed gastric image generation based on soft-label dataset distillation for medical data sharing,” *Computer Methods and Programs in Biomedicine* (IF: 7.0), vol. 227, 107189, 2022.

[**A-3**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “COVID-19 detection based on self-supervised transfer learning using chest X-ray images,” *International Journal of Computer Assisted Radiology and Surgery* (IF: 3.4), vol. 18, no. 4, pp. 715–712, 2022.

[**A-4**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “Boosting automatic COVID-19 detection performance with self-supervised learning and batch knowledge ensembling,” *Computers in Biology and Medicine* (IF: 7.7), vol. 158, 106877, 2023.

[**A-5**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “Self-supervised learning for gastritis detection with gastric X-ray images,” *International Journal of Computer Assisted Radiology and Surgery* (IF: 3.4), pp. 1-8, 2023.

[**A-6**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “Dataset distillation using parameter matching,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 2023. (Accepted for publication)

[**A-7**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “Importance-aware adaptive dataset distillation,” *Neural Networks* (IF: 9.7), 2023. (Under revision)

[**A-8**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "RGMIM: Region-guided masked image modeling for learning meaningful representation from X-ray images," 2023. (Under review)

[**A-9**] R. Togo*, G. Li*, K. Mabe, S. Nishida, Y. Tomoda, T. Ogawa, and M. Haseyama, "Artificial Intelligence-assisted atrophic level classification: comparison with endoscopists and multicenter evaluation," 2023. (* Equal contribution, Under review)

[**A-10**] Y. Gan, G. Li, R. Togo, K. Maeda, T. Ogawa, and M. Haseyama, "Zero-shot traffic sign recognition based on mid-level feature matching," 2023. (Under review)

(B) **International Conference**

[**B-1**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Soft-label anonymous gastric X-ray image distillation," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pp. 305–309, 2020.

[**B-2**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Complexity evaluation of medical image data for classification problem based on spectral clustering," in *Proceedings of the IEEE Global Conference on Consumer Electronics (GCCE)*, pp. 667–669, 2020.

[**B-3**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Cross-view self-supervised learning based on momentum statistics in batch normalization," in *Proceedings of the IEEE International Conference on Consumer Electronics – Taiwan (ICCE-TW)*, pp. 1–2, 2021.

[**B-4**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Triplet self-supervised learning for gastritis detection with scarce annotations," in *Proceedings of the IEEE Global Conference on Consumer Electronics (GCCE)*, pp. 787–788, 2021.

[**B-5**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Self-supervised transfer learning for COVID-19 detection from chest X-ray images," in *Proceedings of the AAAI Conference on Artificial Intelligence Workshops (AAAIW)*, pp. 1–6, 2022.

[**B-6**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, "Self-knowledge distillation based self-supervised learning for COVID-19 detection from chest X-ray images, " in *Proceedings of* *the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1371–1375, 2022.

[**B-7**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “TriBYOL: Triplet BYOL for self-supervised representation learning,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3458–3462, 2022.

[**B-8**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “Dataset distillation for medical dataset sharing,” in *Proceedings of the AAAI Conference on Artificial Intelligence Workshops (AAAIW)*, pp. 1–6, 2023.

[**B-9**] G. Li, R. Togo, K. Mabe, S. Nishida, Y. Tomoda, T. Ogawa, and M. Haseyama, “Clinical-Guided Endoscopic Gastric Atrophy Level Assessment Based on Kimura-Takemoto Classification,” 2023. (Under review)

(C) **Domestic Conference**

[**C-1**] 李広, 藤後 廉, 小川 貴弘, 長谷山 美紀, “nAULSに基づくデータセットの複雑性評価に関する検討,” 令和2年度電気・情報関係学会 北海道支部連合大会, pp. 98–99, 2020.

[**C-2**] G. Li, R. Togo, K. Mabe, S. Nishida, Y. Tomoda, H. Shimizu, T. Ogawa, and M. Haseyama, “A note on automatic diagnosis of Helicobacter pylori infection based on EfficientNet with flooding loss,” ITE Technical Report, vol. 45, no. 4, pp. 23–26, 2021.

[**C-3**] G. Li, R. Togo, K. Mabe, S. Nishida, Y. Tomoda, T. Ogawa, and M. Haseyama, “A note on automatic diagnosis of Helicobacter pylori infection based on self-supervised learning and self-knowledge distillation,” ITE Technical Report, vol. 46, no. 6, pp. 49–52, 2022.

[**C-4**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “COVID-19 detection based on masked image modeling using vision transformer,” Meeting on Image Recognition and Understanding (MIRU), pp. 1–4, 2022.

[**C-5**] 李広, 藤後 廉, 小川 貴弘, 長谷山 美紀, “医療データを対象としたデータセット蒸留に関する検討,” 第1回 北海道大学医療AIシンポジウム, p. 1, 2022.

[**C-6**] Y. Gan, G. Li, R. Togo, K. Maeda, T. Ogawa, and M. Haseyama, “A note on traffic sign recognition based on vision transformer adapter using visual feature matching,” ITE Technical Report, vol. 47, no. 6, pp. 208–211, 2023.

[**C-7**] G. Li, R. Togo, T. Ogawa, and M. Haseyama, “Dataset distillation via self-adaptive parameter matching,” Meeting on Image Recognition and Understanding (MIRU), pp. 1–5, 2023.

(D) **Award**

[**D-1**] IEEE GCCE 2020 Excellent Student Paper Award Gold Prize, 2020.

[**D-2**] 電子情報通信学会北海道支部学生奨励賞, 2022.

[**D-3**] 第1回 北海道大学医療AIシンポジウム優秀研究賞, 2022.

[**D-4**] 学院長賞, 2023.

(E) **Scholarship**

[**E-1**] JGC-S Scholarship, 2021.

[**E-2**] 北大・日立協働教育研究支援プログラム, 2022.

(F) **Media Coverage**

[**F-1**] “How to Make Artificial Intelligence More Democratic,” Scientific Amercian (USA), 2021/01/02. (G. Li et al., ICIP 2020)

[**F-2**] “How to Make Artificial Intelligence More Democratic,” Deccan Herald (India), 2021/01/02. (G. Li et al., ICIP 2020)

[**F-3**] “Top 3% Attention Score Paper,” Altmetric (UK), 2021/01/02. (G. Li et al., ICIP 2020)

[**F-4**] “How to Make Artificial Intelligence More Democratic,” Global Science (China), 2021/01/08. (G. Li et al., ICIP 2020)

[**F-5**] “北海道大 情報科学研究院 AI活用でインフラ点検,” 日本経済新聞, 2021/06/16. (G. Li et al., ICIP 2020)

[**F-6**] “メディア工学の研究動向,” 映像情報メディア年報2021-22シリーズ, 2022/01/01. (G. Li et al., ICIP 2020)

[**F-7**] “医用画像データ共有へ向けたデータ圧縮技術の構築,” 北海道大学病院 医療AI研究開発センター プロジェクト, 2022/03/29. (G. Li et al., ICIP 2020)

[**F-8**] “肺X線画像からのCOVID-19肺炎検出AIの構築,” 北海道大学病院 医療AI研究開発センター プロジェクト, 2022/03/29. (G. Li et al., ICASSP 2022)

[**F-9**] “最先端マルチメディア技術とその実社会応用,” 北海道総合通信局 公式チャンネル, 2022/04/20. (G. Li et al., ICIP 2020; ICASSP 2022)

[**F-10**] “Most Popular AI Research Aug 2022,” LibHunt (USA), 2022/09/03. (G. Li et al., Awesome-Dataset-Distillation)

[**F-11**] “A project to help you understand Dataset Distillation,” Synced (China), 2022/10/09. (G. Li et al., ICIP 2020; Awesome-Dataset-Distillation)

[**F-12**] “Top 10 Self-supervised Learning Models in 2022,” Analytics India Magazine (India), 2022/11/02. (G. Li et al., ICASSP 2022)

[**F-13**] “2022 Top10 Self-supervised Learning Model Released,” AI Era (China), 2022/11/12. (G. Li et al., ICASSP 2022)

[**F-14**] “第1回 北海道大学医療AIシンポジウムを開催しました,” 北海道大学病院 医療AI研究開発センター ニュース, 2022/11/25. (G. Li et al., 第1回 北海道大学医療AIシンポジウム)

[**F-15**] “北大・日立協働教育研究支援プログラム発表会を実施しました,” 北海道大学大学院総合サイト ニュース, 2023/01/06. (G. Li et al., 北大・日立協働教育研究支援プログラム)

[**F-16**] “第1回北海道大学医療AIシンポジウム開催報告,” 北海道放射線医学雑誌, 2023/03/23. (G. Li et al., 第1回 北海道大学医療AIシンポジウム)

(G) **Project**

[**G-1**] G. Li, B. Zhao, and T. Wang, “Awesome-Dataset-Distillation,” 2022/08/07. (800+ Stars)