Автономная некоммерческая организация высшего образования
«Университет Иннополис»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(БАКАЛАВРСКАЯ РАБОТА)
по направлению подготовки
09.03.01 - «Информатика и вычислительная техника»

GRADUATION THESIS
(BACHELOR'S GRADUATION THESIS)
Field of Study
09.03.01 – «Computer Science»

Направленность (профиль) образовательной программы
«Информатика и вычислительная техника»
Area of Specialization / Academic Program Title:
«Computer Science»

Тема / Topic	**Применение техник сравнительного самообучения в задаче классификации рентгеновских снимков грудной клетки / Contrastive self-supervised learning for chest X-ray's classification**

Работу выполнил / Thesis is executed by	**Кивотова Евгения Александровна / Kivotova Evgeniia**	подпись / signature
Руководитель выпускной квалификационной работы / Supervisor of Graduation Thesis	**Протасов Станислав Игоревич / Protasov Stanislav**	подпись / signature
Консультанты / Consultants	**Максудов Булат Тимурович / Maksudov Bulat**	подпись / signature

Иннополис, Innopolis, 2021

Contents

List of Tables

List of Figures

Abstract

For the last decades, people were working on improving medical images analysis with the help of machine learning algorithms. Most of such algorithms are supervised and require a large amount of labeled data to produce accurate results. The lack of available labeled data is a common problem for medical tasks. Therefore Self-Supervised methods that work even with unlabeled images triggered recent research interest. Little work is done on analyzing the impact of Self-Supervised pretraining on the task of chest X-rays classification. In this study, we answer how MoCov2 and SimCLR Self-Supervised pretraining methods affect the classification of chest X-ray images from the CheXpert dataset. To answer this question, we defined two sets of target labels from the CheXpert dataset, applied two different techniques to pretrain the ResNet18 model on ordinary and expanded unlabeled sets of images, and compare fine-tuned models with the performance of the non-pretrained baseline model. Our findings were that MoCov2 pretraining on images from both the CheXpert and the expanded dataset improves the AUROC metric of multi-label classification of CheXpert images using all CheXpert classes. The same pretraining showed no significant effect on the accuracy of CheXpert images classification when only rare classes were considered. In contrast, SimCLR pretraining with strong image augmentations made the model perform worse in the classification task. Obtained results indicate that Contrastive Self-Supervised techniques can improve model performance in chest X-ray classification task with a thoughtful choice of hyperparameters and image augmentations.

Chapter 1

Introduction

Machine learning models need large volumes of labeled data to train from scratch. Unfortunately, not every task is supplied with enough labeled data for model training. The lack of labels is common for the medical field, where data collection is expensive and time consuming because one image requires the consensus of two or three specialists minimum to be labeled.

The general scheme that deals with limited labeled data uses some pretraining technique before targeting the problem of interest. The pretraining methods have a large variety of implementations. One of the most common solutions is to use a model pre-trained in a supervised manner on a large labeled dataset such as ImageNet [1]. Another possibility is to use some self-supervised pretraining technique on available unlabeled data. When the supervised ImageNet pretraining is commonly used for the pretraining stage, even for medical data, the application of self-supervised pretraining in the medical field has had limited attention so far. This lack of attention especially applies to Chest X-ray data.

In this work, we are studying the influence of self-supervised pretraining

on Chest X-ray image classification. Additionally, we are interested in a situation when only a small portion of available training data has accurate labels. Lastly, we focus on the effect of pretraining dataset expansion with data from additional sources but in the same domain.

1.1 Self-Supervised learning

Self-supervised learning is part of the unsupervised learning area in machine learning. The methods in this area do not need labels for data samples. Instead, the labels are created from the training data itself by applying different modification functions to it. Such labels are called *pseudo labels*. Researches use two more essential terms in self-supervision: *Pretext task* and *Downstream task*. The Pretext task represents a problem with artificially created labels (pseudo-labels). These labels are used during model training to learn good representations of objects. The Downstream task evaluates the quality of the received data representations learned during the Pretext task.

Self-Supervised methods for computer vision can be divided into two main categories: Contrastive and Generation-Based. Contractive self-supervised learning methods are based on learning differences between data samples. These methods transform input images into numerical labels and use variations of contrastive loss. In contrast, Generation-Based methods aimed to reconstruct the data sample or its parts. In opposite to contractive methods, they generate a new image from the input picture. This study focuses on methods from the Contractive category because they are straightforward to implement, train, and test.

1.2 Contribution

We highlight the following important contribution points of this study:

- We have formulated a convincing use case of a Self-Supervised learning application for Chest X-ray's classification.
- We proposed image transformation functions and hyperparameters for MoCov2 model pretraining that improves chest X-ray classification.
- We provided a rich literature review of existing studies in this and related areas, adopting best practices for our experiments.
- We performed statistical analysis of our results and formulated our conclusions based on the significance of the outcomes.

1.3 Structure

This study has the following structure. Firstly, Chapter 2 provides a detailed review of the Self-Supervised Learning methods proposed and used in literature. Then Chapter 3 describes the datasets, model and methods that we use for hypothesis testing. Next, implementation and experiment information is provided in Chapter 4. Next, Chapter 5 describes the results, our interpretation of observed outcomes, and shows the discussion around future application and limitations of used methods. Finally, we conclude our study in Chapter 6.

Chapter 2

Literature Review

2.1 Navigation

In this chapter, we present a review of the literature related to the usage of self-supervised learning techniques in a chest x-ray classification problem. In Section 2.2, we define which self-supervised methods have been developed and by which properties they may be grouped. We do it to understand the basics of self-supervised learning. Next, in Section 2.3, we narrow down from the entire field of computer vision to the medical data analysis. There we explore how techniques listed in Section 2.2 have been applied to medical data and what results such experiments produce. Lastly, In Section 2.4, we state common problems that researchers face in chest x-ray analysis, observe solutions used, and define the research gap that we will cover with this study.

2.2 Self-supervised methods

At the beginning of our research, we identified the set of self-supervision techniques that have already been developed to understand their properties

and the type of data they apply to. Jing and Tian [2] created survey on techniques of self-supervised learning for images and videos. The authors divide self-supervision methods for image analysis into two main categories: *Context-Based* and *Generation-Based*. With the continuation of developments and the emergence of more and more studies in the field of self-supervision, the presented classification became outdated and required improvement. Therefore, Liu, Zhang, Hou, *et al.* [3] extended the work done by Jing and Tian and provide a comprehensive analysis of existing self-supervised methods in computer vision, natural language processing, and graph analysis. The authors follow the categorisation made by Jing and Tian, populating classes with more methods related to image analysis, naming first category as *Contrastive* Self-Supervised Learning instead of *Context-Based* and second category as *Generative* Self-Supervised Learning instead of *Generation-Based*. Additionally, they highlight a third, Generative-contrastive or *Adversarial* class. This category is formed by several methods from the *Generation-Based* class. Liu, Zhang, Hou, *et al.* explained their decision by the fact that *Adversarial* methods share properties of both *Contrastive* and *Generative* learning methods and cannot be strictly placed into any of these two categories.

2.2.1 Contrastive Self-supervised Learning

The largest category derived by Liu, Zhang, Hou, *et al.* [3] and Jing and Tian [2] is *Contrastive* self-supervised learning. It contains six different self-supervised methods in total: Relative Position Prediction [4], Image Jigsaw Puzzle [5], Rotation Prediction [6], Cluster Discrimination [7], Instance Discrimination [8] and Mutual Information Maximization [9]. All these methods differ in implementation but all learn features based on context hidden in im-

ages. As Jing and Tian [2] stated, in context-based supervision the pretext tasks for Conv Networks are designed in such way that models learn attributes of the *context* of images as their features.

2.2.2 Generation-Based Self-supervised Learning

The smaller category containing only one method is *Generative*.This category was originally defined by Jing and Tian [2] as *Generation-Based* class with lager number of methods inside. But after renaming and regrouping done by Liu, Zhang, Hou, *et al.* [3], only one method related to image analysis remains in this category: Next Pixel Prediction [10]. Jing and Tian provided the common property of methods in this group noting that the current image generation-based methods use pretext tasks that make Conv Networks learn visual features through restoration of deleted or modified image parts.

2.2.3 Adversarial Self-supervised Learning

The last category, *Adversarial*, consists the following four methods: Image Reconstruction [11], Image Colorization [12], Image Inpainting [13] and Super-Resolution [14]. This category is mentioned only by Liu, Zhang, Hou, *et al.* [3]. However, all of these methods appear in *Generation-Based* class derived by Jing and Tian [2]. Liu, Zhang, Hou, *et al.* explained their decision to create additional category by the fact that although the methods they chose are desined for Conv Networks with encoder and a decoder in their structure, decoder component learn image features by retrieving their context attributes.

2.3 Self-supervised Learning and Medical Data

After obtaining the full scheme of self-supervised methods and understanding their categorization, we searched for their application on medical images, which is a more narrow field than general computer vision and closer to our point of interest. We have concluded that self-supervised learning has become increasingly common in research related to medical data analysis. Tajbakhsh, Hu, Cao, *et al.* [15] explained the reasons behind such trend. Using Rotation and Colorization techniques on several types of medical images, they concluded that pre-training models on the medical domain with self-supervision is more effective than transfer learning from an unrelated domain. This study showed that unlabeled data may be involved in model training and improve model performance. Subsequently, the large survey of methods to handle imperfect medical image datasets appeared in the study of Tajbakhsh, Jeyaseelan, Li, *et al.* [16]. The authors mentioned self-supervised pre-training as one of the proactive methods and provided references to different studies that apply self-supervision on various types of medical images. Chen, Bentley, Mori, *et al.* [17] introduced the context restoration self-supervision method and applied it on brain Magnetic Resonance (MR) images, abdominal Computed Tomography (CT) images, setal Ultrasound (US) images, and other medical datasets, at the same time claiming that their method may be used in any domain of 2D images. The application of model pre-training with positive/negative pairs generation on Spinal Magnetic Resonance Images (MRIs) appeared in the study made by Jamaludin, Kadir, and Zisserman [18]. Both studies found that the self-trained CNN required much fewer annotated training samples than the network trained with randomly initialized weights to achieve compatible performance.

2.4 Chest X-ray analysis

The chest X-ray (CXR) images is an important part of the medical data field. CXR analysis using techniques of Computer Vision is one of the trending fields of study in the Data Science community. With an emergence of large labeled CXR datasets such as ChestX-ray14 [19], MIMIC-CXR [20] and CheXpert [21], the development and testing various methods of CXR analysis became accessible to the public. Bressem, Adams, Erxleben, *et al.* [22] examined fifteen different artificial neural network designs that had been pre-trained on ImageNet [1] dataset to find the best models for deep learning tasks on chest radiographs in terms of training time and performance. The authors concluded that considering both AUROC and AUPRC metrics, VGG-16 [23], ResNet-34 [24], and AlexNet [25] achieve the best performance in the CXR classification task. At the same time, Lenga, Schulz, and Saalbach [26] reported the problem of significant domain bias in different public x-ray datasets caused by such factors as hospital-specific processes, the patient population, data collection strategy, and the employed labeling techniques. Because of such bias shift, models trained on one dataset are less accurate when tested on the other datasets. Lenga, Schulz, and Saalbach tested several transfer learning approaches and showed that after adapting to the target dataset with Learning without Forgetting (LwF), performance on the source dataset slightly improves without any source data being involved in adaptation. However, transfer learning does not answer the question of how large volumes of unlabeled data may be used to improve model performance.

2.5 Conclusion

Although various studies related to the effect of self-supervision pre-training have appeared in the field of medical image analysis, a limited number of them explore the applicability of such pre-training techniques in the CXR classification task. The closest type of data that appeared in the studies we found were chest CT. Therefore in the current study, we will focus on the CXR domain and will compare self-supervised pre-training techniques applied to the CXR classification task.

Chapter 3

Methodology

3.1 Navigation

This section fully describes the methods that we used in our study. Section 3.2 lists datasets for our methods and short information about each of them. Then, in Section 3.3 we explain the purpose of the Pretext task in Self-Supervised learning. In addition, we provide brief description of used Pretext tasks, which are MoCov2 (Section 3.3.1) and SimCLR (Section 3.3.2). Next, Section (3.4) defines our Downstream task, which is the second and last task in Self-Supervised learning methods. Final model architectures and differences between them we specify in Section 3.5. We evaluate these models using pipelines and metrics presented in Sections 3.6 and 3.7.1 accordingly. Lastly, Sections 3.7.2 describe general detail about statistical analysis applied to evaluate observed outcomes. data processing and model learning.

3.2 Datasets

Datasets choice plays the most important role in our study. When choosing datasets, we rely on two important factors. Firstly, to make our results reproducible, we mostly focus on using public chest x-ray datasets. Secondly, we take the most popular datasets to compare our findings with existing solutions and to make it easier for other researchers to compare their results with ours. Given the said factors, we collected CheXpert [21], ChestX-ray14 [19], Chest X-Ray Pneumonia [27], GB7-FLG, TBX11K [28], and Vinbigdata [29] datasets for our experiments. The following subsections contain a full description of each dataset.

3.2.1 CheXpert

CheXpert is a large collection of chest X-ray images labeled for the presence or absence of several diseases [21]. The dataset consists of 224 316 chest X-rays collected from 65 240 patients. Each image is associated with 14 different labels. Every label can have one of four possible values: null value for mention absence, 1 if disease is present, 0 if disease is not detected and -1 if the disease presence is uncertain. Due to a large number of label categories it is possible that one image can be associated with several positive labels.

3.2.2 ChestX-ray14

ChestX-ray14 is the extension of ChestX-ray8 dataset [19]. This extension adds six new label categories expanding the categories number from 8 to 14. The whole dataset contains 108 948 frontal-view X-ray images of 32 717 unique patients with text-mined disease image labels. Unlike CheXpert dataset, Wang,

Peng, Lu, *et al.* used only two different values for labels: 0 if disease is not detected and 1 if disease is detected. Multiple positive labels can be connected into one image, similarly to CheXpert.

3.2.3 Chest X-Ray Pneumonia

The Chest X-Ray Pneumonia dataset [27] comprises 5 856 chest X-ray pictures from children and adults, with 3,883 classified as indicating pneumonia (2 538 bacterial and 1 345 viral) and 1 349 classified as normal.

3.2.4 GB7-FLG

GB7-FLG is the dataset, received by Laboratory of Artificial Intelligence in Innopolis University from Polyclinic department of Kazan City Clinical Hospital No. 7. It is small dataset of 127 images, including 28 images with pneumonia and 99 images of normal lungs.

3.2.5 TBX11K

Tuberculosis X-ray (TBX11K) [28] is a large-scale dataset for Tuberculosis detection on CXR images. This dataset contains 11200 CXR images or four classes: healthy, sick without tuberculosis, active tuberculosis, latent tuberculosis, both active and latent tuberculosis.

3.2.6 Vinbigdata

Vinbigdata dataset was released in January 2021 by Vingroup Big Data Institute. It contains 18 000 CXR images with 6 global labels and 22 local

labels. More information is available in the paper of Nguyen, Lam, Le, *et al.* [29].

3.3 Pretext task

The pretext task combines the method for pseudo label mining from unlabeled data and the subsequent task to learn the relation between data samples and mined labels. In addition, it is used to learn general data representations and get model weights more adapted to the target data domain.

The pretext task in Contrastive Self-Supervised Learning employs a discriminative technique in embedding space to group similar data closer together and different ones far apart. It is done with the help of contrastive loss that shows how close data embeddings are. In computer vision tasks, the encoder network produces feature representations of the images. The typical way to produce a positive sample for each image is to apply a set of data augmentations on that image and consider its embedding a positive example. On the opposite, the embedding of any other image from the dataset is considered a negative sample.

In this study, we decided to analyse the effect of next model pretraining on the downstream task: MoCov2 [30] and SimCLR [31]. We have chosen these pretraining methods because they produce results comparable to the state-of-the-art supervised method on ImageNet dataset [1] according to Jaiswal, Babu, Zadeh, *et al.*

3.3.1 MoCov2

Method The second version of the Momentum Contrast algorithm [30], referred to as MoCov2, uses instance discrimination through dictionary look-up task to learn image representations. This task considers each image as a query. The query is studied against a set of keys to identify the most similar pair. By keys, MoCov2 defines all positive and negative images from its internal queue. The loss function forces positive keys to be close to the query while making negative keys as different as possible. To create a set of queries and keys, MoCov2 uses two views of each image by applying a stochastic data augmentation function twice to each image. Given images x_i and x_j sampled from the dataset X, three independent transformations t_1,t_2 and t_3 from a distribution of data augmentation functions T, query encoder model e_q and key encoder model e_k, one query is defined as $q_i = e_q(t_1(x_i))$, positive key for it as $k_i = k_+ = e_k(t_2(x_i))$ and one of negative keys as $k_j = e_k(t_3(x_j))$. MoCov2's general novelty is in using a dynamic queue for storing and updating the keys in the dictionary, which allows testing each query against a rich number of negative samples.

Loss Function The authors of the MoCov2 used InfoNCE contrastive loss function in their paper [30]. In this function (3.1), τ is a temperature hyper-parameter, k_+, q and k – are vector representations, described in Section 3.3.1. InfoNCE is similar to classic Categorical Cross-Entropy loss (3.2). The intuition behind InfoNCE loss is to classify q as k+ in softmax-based classifier manner. The dot product of two vectors works similar to a binary value. It is positively large for «similar» vectors and becomes negative for absolutely «different» vectors.

$$L_q = -log\frac{exp(q \cdot k_+/\tau)}{\sum_{i=0}^{K} exp(q \cdot k_i/\tau)} \tag{3.1}$$

$$L = -\sum_{i=1}^{C} y_i log\frac{exp(\hat{y}_i)}{\sum_{j=1}^{C} exp(\hat{y}_j)} \tag{3.2}$$

3.3.2 SimCLR

Method SimCLR is the second Contrastive Self-Supervised method of model pretraining used in our study. The general idea behind this method is simmilar to MoCov2, described earlier in Section 3.3.1. Similarly, it produces two differently augmented images from one input image and maximizes agreement between images in such pairs via a contrastive loss to learn image representations. Given image x_i from dataset X, independent transformations t_1 and t_2 from a distribution of data augmentation functions T, encoder network f and non-linear projection network g, the positive pair of image representations are defined by $z_i = g(f(t_1(x_i)))$ and $z_j = g(f(t_2(x_i)))$. In such a way, each batch of N input examples is transformed into batch of $2N$ examples.

Loss function The loss function (3.3) for representations z_i and z_j resulted from one image x_i in SimCLR method is similar to MoCov2 loss function (3.1), but it has two diffeerences. First difference relates to the sum in the denominator. In MoCov2 it is done across dictionary with K keys, when in SimCLR is is calculated over batch of $2N$ images. This difference implies that SimCLR requires larger batch size than MoCov2. The second difference is in similarity function. MoCov2 uses dot product, whereas SimCLR has $sim(z_i, z_j) = {z_i}^T \cdot z_j/(\| z_i \| \cdot \| z_j \|)$, which is cosign similarity formula.

$$L_q = -log\frac{exp(sim(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{(k\neq i)} exp(sim(z_i, z_k)/\tau)} \tag{3.3}$$

3.4 Downstream task

The Downstream task is the real problem that must be solved using provided data and labels. It can be any task for supervised learning, for example, classification, detection, segmentation, and others.

We decided to use the disease presence detection in images from the CheXpert dataset as the Downstream task. Each disease is handled separately in such a task and is classified in a binary manner as detected or not detected.

According to our research question, we are interested in model performance on the downstream task in the full-dataset setting and the setting of a limited amount of labels available for each class. We found the artificial reduction of labeled data unnatural and not a practical problem. Looking for more natural approach, we, in contrast to resent similar studies ([33], [34]), focused on rare diseases classification.

3.4.1 Full-dataset setting

The idea behind the full-dataset setting is simple. This setting assumes that we have a label for each data sample and are trying to improve the classification metrics with Self-Supervised pretraining. For this purpose, we use all Frontal view images from the CheXpert dataset during classification training.

3.4.2 Rare diseases setting

Another setting we are interested in is when not all available data samples have labels. For example, there exist a study performed by Sowrirajan, Yang, Ng, *et al.* [33] that analyses the performance of CheXpert images classification with and without model pretraining with the MoCov2 algorithm. The authors indicate visible improvement of classification when only a tiny fraction of labeled data is used for the downstream task, but we found their method of artificial label shortage not practical. Our study seeks the natural problem of label shortage and has found it in rare diseases. Such diseases naturally appear in a small number of medical cases. The limited occurrence produces the limited labeled data and the extraordinary complexity of such case identification.

The relatively big size of the CheXpert dataset allows us to see the quite representative distribution of disease occurrence in the concrete data source. It might not show the actual situation in the whole world, but suitable for the concrete region. To identify rare diseases, we must first define the method to calculate their distribution.

As the CheXpert may contain several labels per image, we count the number of positive samples for each label separately. One limitation of the CheXpert dataset is that some images may have uncertain label value, and authors provide the analysis of better ways to handle them in a binary classification setting only for five out of fourteen classes. For simplicity, we understand uncertain labels as detected and the absence of any label as not detected.

From resulted distribution shown in Fig. 3.1 we saw the division of classes into two categories: classes with more than 25 000 positive samples and classes with less than 25 000 positive samples. The 25 000 positive samples threshold allows keeping the relatively large subset with an average balance of class sizes.

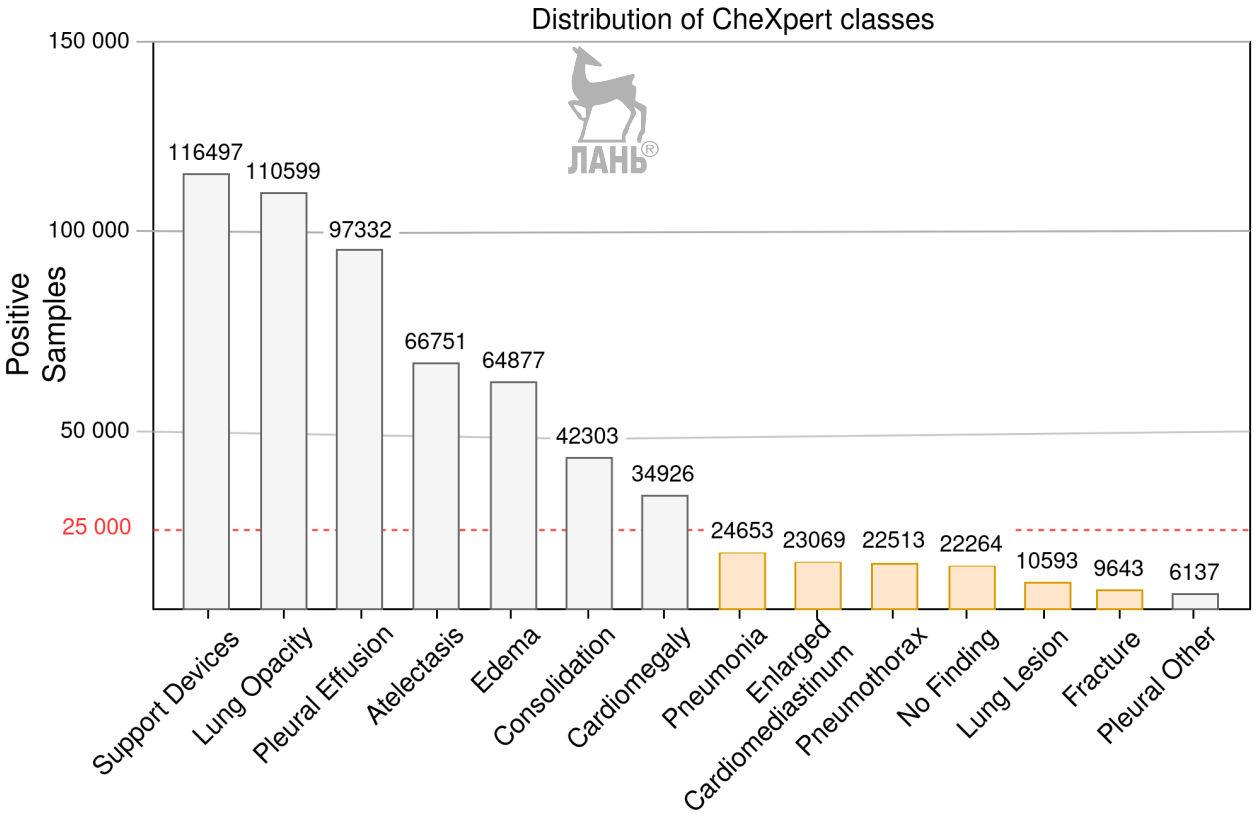


Figure 3.1: The distribution of positive samples over classes in CheXpert dataset.

We analyzed the labels from the last category and defined the following rare diseases: Enlarged Cardiomediastinum, Lung Lesion, Pneumonia, Pneumothorax, and Fracture. We omitted Pleural Other as a separate class because it does not represent factual findings but the group of marginal findings. For a more natural classification of the five mentioned classes, we also considered samples with a positive No Finding label and added this class to the set.

In such a way, we formed a final dataset of 82 942 samples from frontal images with at least one positive label for the six mentioned classes. To handle other labels in the resulting set, we consider any positive label for the other eight classes as a positive label for the additional Other class. The absence of positive labels in all other eight classes results in a negative label for the Other class. Further, we refer to the new dataset as CheXpert-Rare. The high

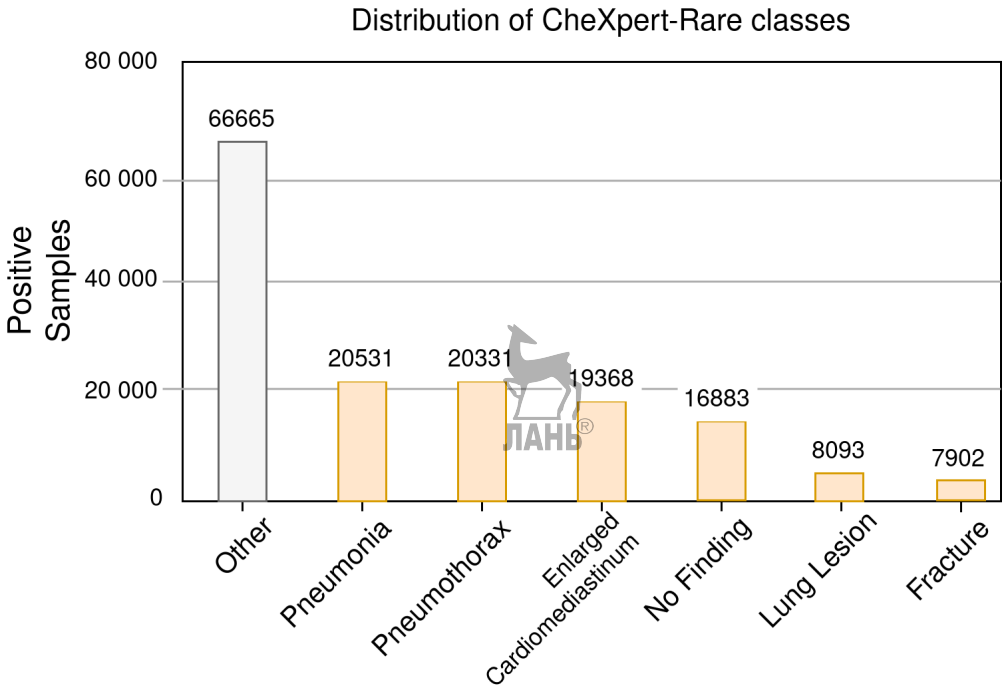


Figure 3.2: The distribution of positive samples over classes in CheXpert-Rare dataset. High amount of positive Other labels indicate that rare diseases are likely to be impaired with more common findings.

amount of positive Other labels compared to the distribution of other classes shown in Fig. 3.2 indicate that rare diseases are likely to be impaired with more common findings in CheXpert images. The minor difference between the sum of positive samples for Enlarged Cardiomediastinum, Lung Lesion, Pneumonia, Pneumothorax, Fracture, No Finding classes and the total amount of images shows the minimum overlap between all these classes.

3.4.3 Loss function for Downstream task

In both settings, described in Sections 3.4.1 and 3.4.2, our Downstream task is a multi-label classification problem, meaning that we had to predict the presence of pathology on the image separately for each class. Formally, multi-label classification is identical to several simultaneous binary classifications.

Therefore, Binary Cross-Entropy (BSE) loss is suitable for such a problem. Given the correct label y_i and predicted probability $\hat{y}_i$ of class presence in an image, we computed the loss function for class i in the following way:

$$BSE(y_i, \hat{y}_i) = -y_i log(\hat{y}_i) - (1 - y_i) log(1 - \hat{y}_i) \tag{3.4}$$

Given C classes, we computed the total loss L_D for the Downstream task as a mean of their BSE values:

$$L_D = \sum_{i=1}^{C} BCE(y_i, \hat{y}_i) \tag{3.5}$$

3.5 Models

For our experiments we use ResNet18 [24] model. Such model choice is based on the fact that it shows one of the best AUROC metric scores in the CheXpert images classification task, being smallest. Therefore the quickest to train from all tested ResNet versions [22]. Moreover, it was already used in a similar study [33]. For both Downstream and Pretext tasks, we use ResNet18 architecture with an original encoder and custom classifier. Architectural schemes, presented in Fig. 3.3, show that the only thing that we change in the ResNet18 classifier for the Downstream task is the size of the output vector. We make it equal to the number of classes in the used dataset: seven for CheXpert-Rare and fourteen for full CheXpert. The change is similar for the MoCov2 Pretext task, but the number of output features is fixed to 128, independently of the data they use. In contrast to mentioned modifications, for the SimCLR Pretext task, we substitute the ResNet18 classifier with a projectile head, consisting of Fully Connected 512 to 512 layer, Batch Normalisation,

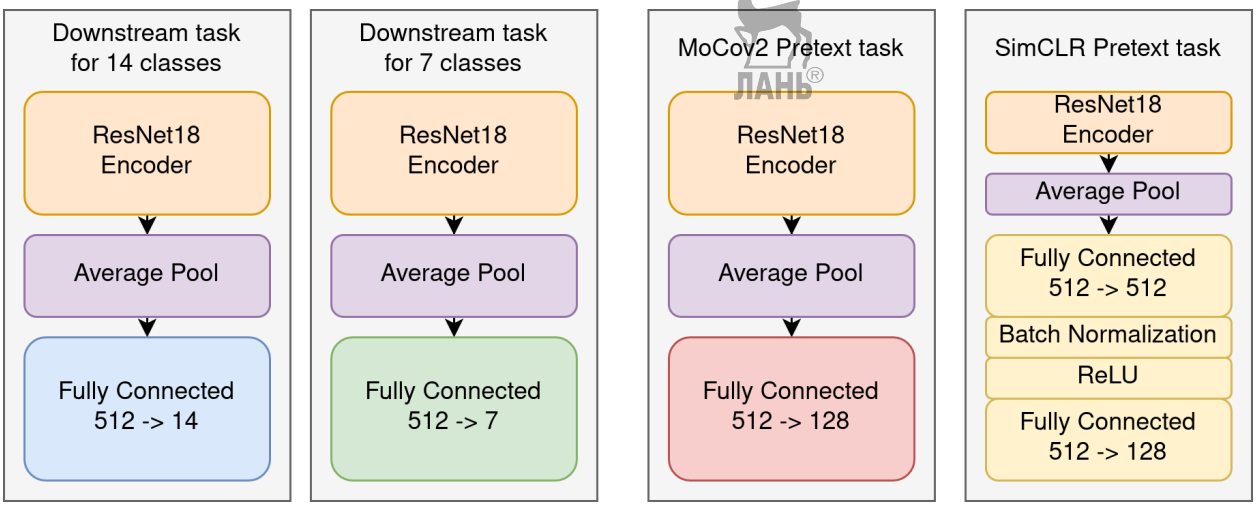


Figure 3.3: The modifications of ResNet18 model for Downstream and Pretext tasks. The Downstream task modifications depend on training data, whereas the Pretext task modifications are fixed for any data used in training.

ReLU activation and final Fully Connected 512 to 128 layer. The last modification does not depend on data as well.

3.6 Pipelines for feature evaluation

To ensure the scientific value of our study results, we had to choose the correct method of feature evaluation. Our study aimed to measure the effect of Pretext task application on model performance in Downstream task. Therefore we had to compare models trained with two different pipelines. The first pipeline consists of our Downstream task, for instance, supervised model training to predict image classes. The second pipeline combines one of our Pretext tasks, namely model pretraining with unsupervised manner, and subsequent model fine-tuning on previously mentioned Downstream task. The schema of both pipelines presented in Fig. 3.4 indicates that both of them use the same set of labeled data for the Downstream task when the second pipeline additionally uses a set of unlabeled data independently from the first pipeline.

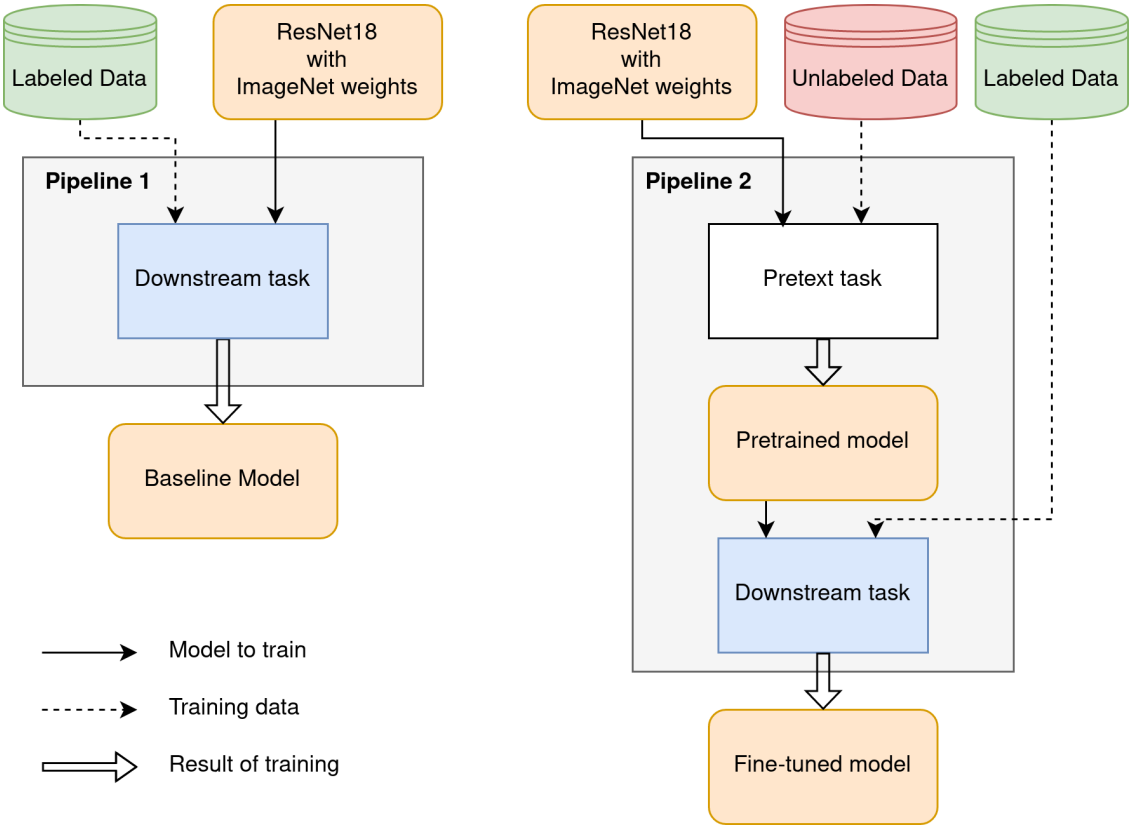


Figure 3.4: The schema of two pipelines, used to evaluate the impact of Pretext task on Downstream task. The key idea of figure is to show that both pipelines use same Labeled data for Downstream task, when the choice of Unlabeled data in the second pipeline is independent from first pipeline.

3.7 Model Evaluation

3.7.1 AUROC metric

We evaluate our models using the AUROC metric. AUROC, known as Area Under the Receiver Operating Characteristics, is a performance measurement for multi-class classification problems. It unites two separate metrics. The first metric is Receiver Operating Characteristics (ROC), representing a probability curve for binary classification for different threshold values. The second metric is Area Under The Curve (AUC) which measure the separability of the model. It shows if the model can distinguish between classes. With the growth of the AUC, the model better predicts 0s as 0s and 1s as 1s. Therefore, having AUROC value ranging from 0 to 1, we are looking for solutions that result in AUROC close to 1.

3.7.2 Statistical analysis

We use non-parametric bootstrap method to estimate how variable is the model performance. To collect estimate distribution for each model, we perform tests on 500 bootstrap replicates drawn from the test set. We then sort resulted values and compute 95% Confidence Intervals by taking 0.025 and 0.975 quintiles of resulted distribution to have significance level at the $p = 0.05$.

3.8 Conclusion

In this section we fully described the methods that we used in our study. We presented full list of obtained datasets, defined the meaning of Pretext and Downstream tasks in Self-Supervised learning. In addition, we provided brief

description of used Pretext tasks, which are MoCov2 and SimCLR. In the end of this section, we illustrated the differences in used model architectures, pipelines and metrics used for model evaluation.

Chapter 4

Implementation and Results

4.1 Navigation

In this chapter, we describe how our experiments were implemented and present the results of our study. First of all, Section 4.2 specifies the general implementation details, such as programming language, machine learning framework, hardware, and hyperparameters used. Section 4.3 presents our image transformation algorithms and explains how we designed them. Finally, in Sections 4.4 and **??**, we fully describe conducted experiments and show how we chose the best models for evaluation.

4.2 General implementation details

We implemented our experiments using Python 3 programming language and PyTorch Lightning [35] framework. For Downstream task, we used ResNet18 model with ImageNet [1] weights from Torchvision[1] package. We used MoCov2 and SimCLR models from PyTorch Lightning Bolts [36] package and

[1] https://pytorch.org/vision/stable/index.html

Table I: The values of learning rate and batch size, used for Downstream and different Pretext tasks.

Task	Learning rate	Batch size
Downstream	3×10^{-5}	32
MoCov2	1×10^{-4}	32
SimCLR	0.5	32

slightly changed their initialisation to make sure they use the same pretrained ResNet18 Encoder as our Baseline model. It was done to meet structural requirements showed in Fig. 3.3. During training, we used Nvidia GeForce GTX 1080 video card. It took about 11-12 hours for this video card to process one training epoch on the full CheXpert dataset.

4.2.1 Train, validation and test splits

In the CheXpert dataset, we divided the official train set into 80% train and 20% validation partitions. Additianally, we used official validation set containing 202 images with manually assigned labels as test data. We did not take images from the train partition to populate this test set with more data because labels in train data are unreliable. Moreover, having no positive samples for Fracture class in the test set, we ommited this class during evaluation. For all other datasets, we used 80% of all data for train partition and 20% - for validation. Since models are evaluated only on CheXpert data, these datasets are not required during the test stage. In all splits, the label distribution was the same as in the original datasets.

4.2.2 Hyper Parameters

Models have a large set of hyperparameters. This set contains general parameters, such as a learning rate, batch size, and individual parameters. Table I shows what values of learning rate and batch size we used in our experiments. For the Downstream task, we followed the original CheXpert paper [21], but the learning rate of 1×10^{-4} used by the authors made our Baseline model overfit right after the first epoch of training. Therefore, for the Downstream task, we reduced the initial learning rate to 1×10^{-5}. For the MoCov2 Pretext task, we used an initial learning rate of 1×10^{-4} that authors used in their paper [30]. The same was applied to the SimCLR Pretext task - we set the initial learning rate to 0.5 following the paper [31] of this algorithm.

The batch size parameter was restricted only by the throughput of our storage machine. Although the SimCLR algorithm requires a large batch size, we set this parameter to 32 for all experiments. While processing batches of such size, storage was working stably without a dramatic increase in the training time.

Due to time constraints, all Downstream task training runs were restricted by a maximum of 20 epochs of training. Some of such training runs were stopped even earlier due to overfitting or too small learning rate. In contrast to the Downstream task, MoCov2 and SimCLR task runs were not restricted by the number of epochs.

4.2.3 Schedulers and Checkpoints

Schedulers and Checkpoints are helpful techniques that make the model training more stable and protected from unexpected shutdowns of hardware.

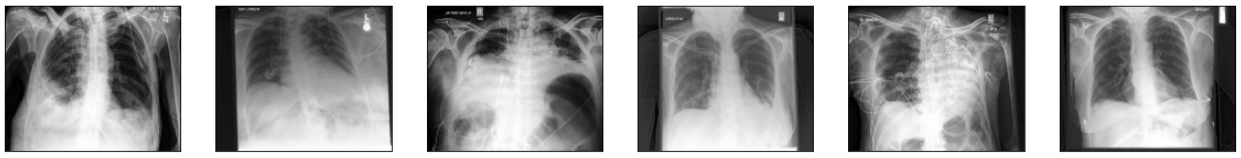
Schedulers control the value of the learning rate, reducing it if a certain event appeared. During Downstream task training, we multiplied the learning rate by 0.1 factor if the value of validation loss did not update its minimum in three subsequent epochs. For Pretext task models, we used schedulers that are built in their implementation inside the Pytorch Lightning Bolts package [36].

Checkpoints are helpful to save the best model weights continuously. Moreover, they are used to resume the model training in the case of its unexpected stop. In all our experiments, we continuously save the model only if it updates the minimal value of validation loss.

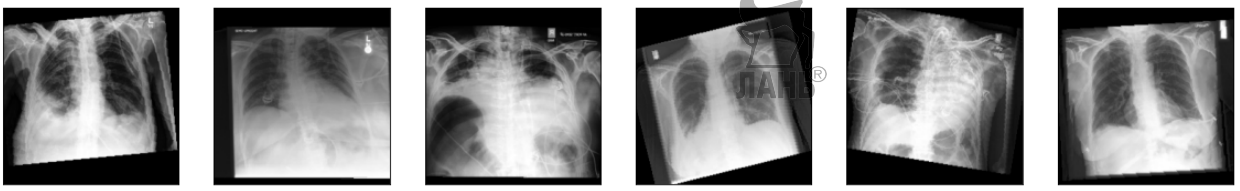
In addition to automatic callbacks, we manually stopped training in cases when the learning rate became lower than 1×10^{-7}. This fact which is noticeable in Fig. 4.3 and Fig. 4.4, where Baseline models were stopped before finishing 20^{th} epoch.

4.3 Image Preprocessing

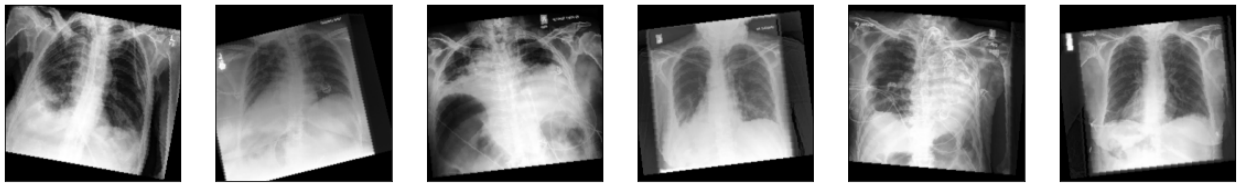
Image preprocessing is an important stage of model training because it largely affects the final model performance. Therefore, an image transformation algorithm must be designed deliberately for each training task. Sequences used for our Downstream and Pretext tasks in existing papers [22], [34], [33] differ from one study to another. Therefore, we designed a custom sequence of transformations based on common augmentations used in each approach and our data specification. We decided to correct the common sets of transformations because CXR images are sensitive to several types of augmentation methods, such as Gaussian blurring, Noise addition, Scaling, Translation, and Cropping. All these augmentations can corrupt important parts of medical images, erasing



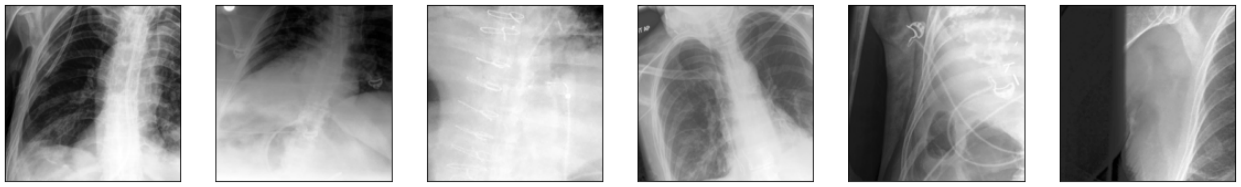
(a) Examples of original images from CheXpert dataset



(b) Examples of transformed images from CheXpert dataset for Downstream task



(c) Examples of transformed images from CheXpert dataset for MoCov2 Pretext task



(d) Examples of transformed images from CheXpert dataset for SimCLR Pretext task

Figure 4.1: The change of original images (a) from CheXpert dataset after transformation for different tasks (b,c,d).

evidence of disease from it. We did not use Gaussian blurring with Noise addition and set all other transformations from the list to have minimal effect to avoid such unwanted image corruption. Before all transformations, we turned each source CXR image from the grayscale into RGB by stacking a given channel with its two copies. We did it because our model can process only RGB images. The following sections contain full details of subsequent image transformations for each of our tasks, and Fig. 4.1 shows how images changed after transformations.

4.3.1 Image Preprocessing for Downstream task

We used the same sequence of augmentation functions for the Downstream task in all experiments. This sequence consists of:

1. **Square Padding** -We padded images with zeros to equal resolution in height in width to save their original proportions after subsequent transformations.

2. **Resize** - The images were resized to the resolution of 224×224 pixels following the official torchvision guidelines [2] for using ImageNet [1] pretrained model.

3. **Random Rotation** - We rotated the images by the random angle from -15 to 15 degrees.

4. **Random Translation** - The images are shifted by height and width axes. The width axes shift d_x is sampled from range $-224 \times 0.05 \leq d_x \leq 224 \times 0.05$, the height axes shift d_y – from range $-224 \times 0.05 \leq d_y \leq 224 \times 0.05$.

5. **Random Scale** - We scaled the images by a random factor, sampled from range $[0.95, 1.05]$ with resolution preserving.

6. **Random Horizontal Flip** - The images were horizontally mirrored with a 50% chance.

7. **Normalisation** - We normalized final RGB images to have the same mean and standard deviation of pixel values as ImageNet [1] images have, according to torchvision guidelines [3].

[2] https://pytorch.org/hub/pytorch_vision_resnet/

[3] See footnote 2

For validation stage, we excluded random functions from set.

4.3.2 Image Preprocessing for MoCov2 Pretext task

For the MoCov2 pretext task, we used the same set of augmentations as for the Downstream task. This decision applied to both training and validation stages. Having no available studies on how different transformations affect the quality of model pretraining after the MoCov2 task is performed on CXR images, we had no reason to change the set of augmentations chosen above. The only difference from the Downstream task augmentations is that we used random transformations during the validation stage. It must be done because the model requires both images in positive pair to differ from each other. Otherwise, it will be too simple for the model to find the positive key for a query.

4.3.3 Image Preprocessing for SimCLR Pretext task

The authors of the SimCLR algorithm [31] compared different combinations of augmentation functions and finalised that rotation and colour jitter is the best pair for their task. We discarded this option because our images are in grayscale. Instead, we picked the combination that fit our data specifications, which we discussed at the beginning of Section 4.3, and perform well according to SimCLR paper [31]. The final list of transformations includes:

1. **Random Resized Crop** - We randomly cropped square part of image and resized it to 224 × 224 resolution, following the official torchvision guidelines[4] for using ImageNet [1] pretrained model.

[4]See footnote 2

2. **Random Rotation** - We rotated the images by the random angle from -20 to 20 degrees.

3. **Random Horizontal Flip** - The images were horizontally mirrored with a 50% chance.

4. **Normalisation** - We normalized final RGB images to have the same mean and standard deviation of pixel values as ImageNet [1] images have, according to torchvision guidelines [5].

Similar to MoCov2 image transformation, we used the same augmentations list during the validation stage as for the training stage.

4.4 MoCov2 Experiments

To test whether the self-supervised pretraining with MoCov2 Pretext task affects the performance of the Baseline Model on the Downstream Task, we have conducted four experiments: ME1, ME2, ME3, ME4. In each experiment, we ran two pipelines, described in Section 3.6, and compared the performance of the resulted models on test data of the Downstream task. While using MoCov2 as a Pretext task in the second pipeline from Fig. 3.4, we were changing the combination of Labeled and Unlabeled datasets from one experiment to another. For the Labeled dataset, we selected between the full CheXpert dataset and our CheXpert-Rare version of it. For the Unlabeled dataset, the choice was between full CheXpert and the combination of six datasets described in Section 3.2: CheXpert [21], ChestX-ray14 [19], Chest X-Ray Pneumonia [27], GB7-FLG, TBX11K [28], and Vinbigdata [29]. Further, we refer to this combination

[5] See footnote 2

Table II: The combination of datasets for each MoCov2 experiment and the event of interest.

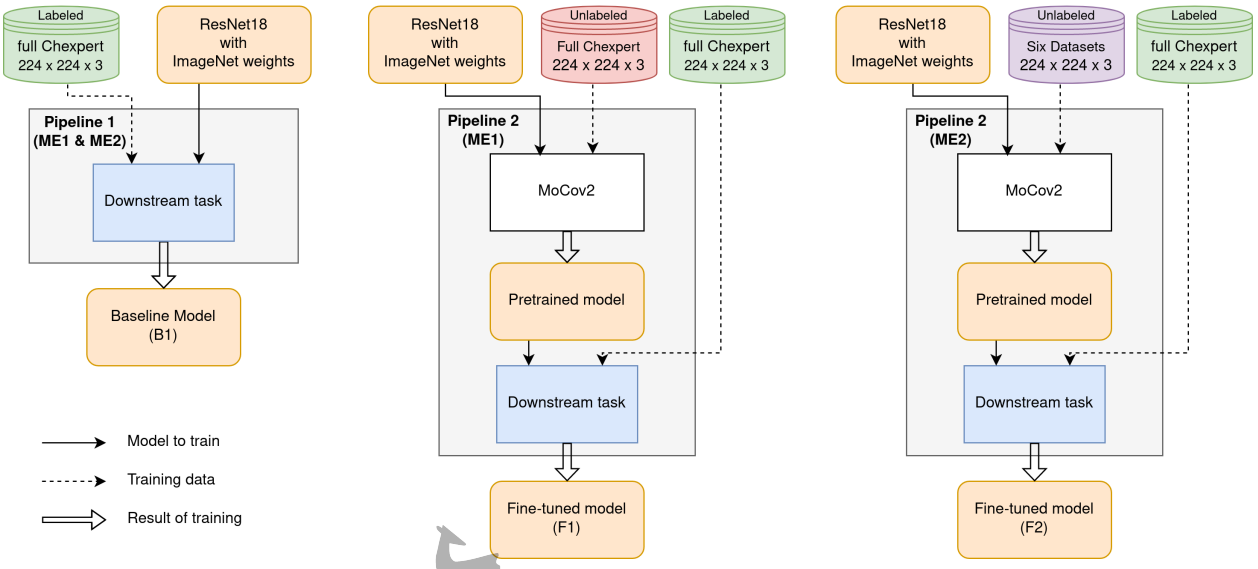
Experiment	Unlabeled data	Labeled data	Studied event
ME1	full CheXpert	full CheXpert	Pretext task addition
ME2	Six datasets	full CheXpert	Unlabeled data expansion
ME3	full CheXpert	CheXpert-Rare	Pretext task addition
ME4	Six datasets	CheXpert-Rare	Unlabeled data expansion

as Six Datasets.

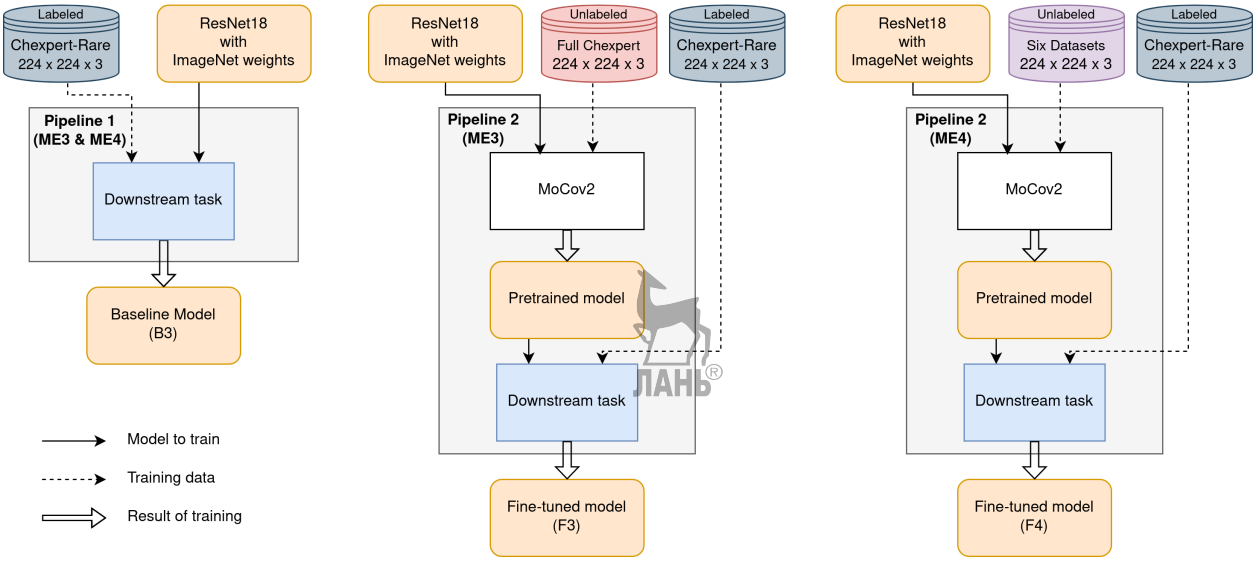
Table II links together the combination of datasets with experiment name. Moreover, it shows our intention to analyze the effects of two events. For the first event, we added MoCov2 pretraining before the Downstream task and studied how this affects the Baseline model performance. Formally speaking, we compared Pipeline 1 with Pipeline 2 on the same dataset. For the second event, we expanded the Unlabeled dataset with additional data from the same domain but from different sources and analyzed the change in pretraining effect. It means that we compared the results of running Pipeline 2 with ordinary and expanded Unlabeled datasets. In the following sections, the result of Pipeline 1 is called Baseline models, whereas the result of Pipeline 2 is called Fine-tuned model.

4.4.1 ME1 and ME2 experiments

In both ME1 and ME2 experiments, we used the full CheXpert dataset for the Downstream task. The detailed scheme of pipelines on Fig. 4.3b indicates the only difference between them: ME1 uses full Chexpert for Pretext task, whereas in ME2, we chose Six Datasets for this stage of Pipeline 2. The following text describes each experiment's purpose, the epoch when each model



(a) Pipelines of ME1 and ME2 experiments. The Pipeline 1 is same for both of them, because we used full CheXpert as Labeled dataset in both experiments. Pipelines 2 differ in Unlabeled data choice. In ME1 we used full CheXpert, when in ME2 we chose Six Datasets.



(b) Pipelines of ME3 and ME4 experiments. The Pipeline 1 is same for both of them, because we used CheXpert-Rare as Labeled dataset in both experiments. Pipelines 2 differ in Unlabeled data choice. In ME3 we used full CheXpert, when in ME4 we chose Six Datasets.

Figure 4.2: The pipelines of all MoCov2 experiments. Experiments are combined together by Labeled dataset that we used in them. Impaired experiment differ in Unlabeled data.

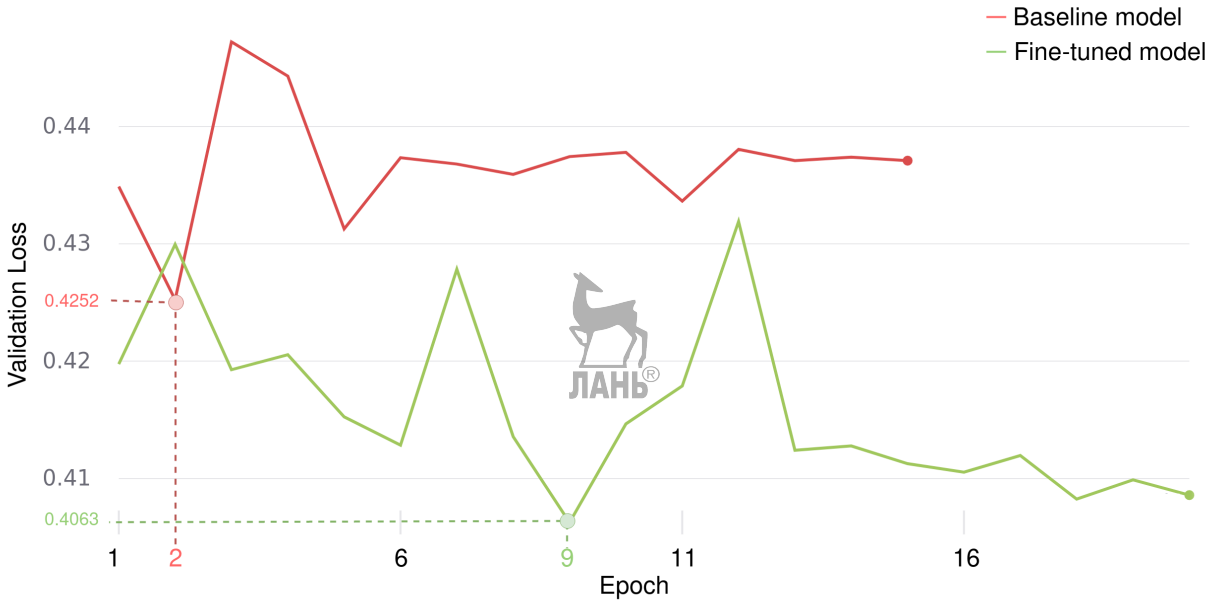
was saved, and the outcomes of each experiment.

ME1: Best checkpoints. We conducted a ME1 experiment to see how the Pretext task affects the model performance on the Downstream task. We wanted to compare the Baseline model and Fine-Tuned model performances when the same data was used during Downstream task training. We performed MoCov2 Pretext task training on the full CheXpert dataset for 12 epoch and stopped due to a small validation loss change compared with previous epochs. According to Fig. 4.3a, the Baseline model reached the best value of validation loss of 0.4252 on 2^{nd} epoch during Downstream task training. At the same time, the best validation loss for the Fine-tuned model was detected on 9^{th} epoch of Downstream task training and had a value of 0.4063.

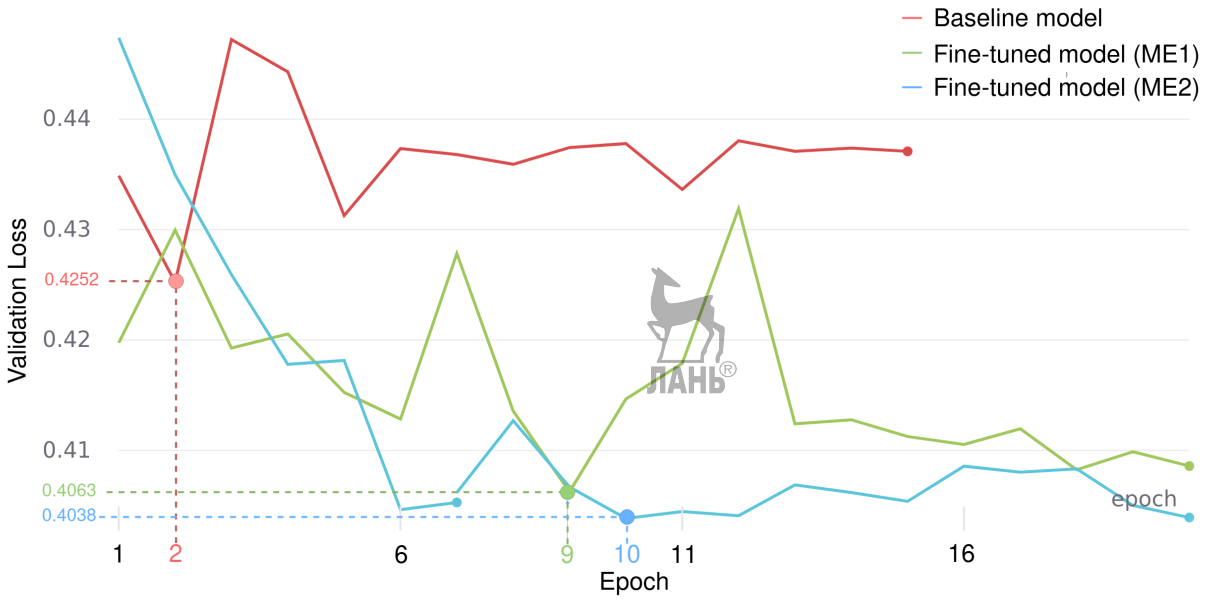
ME2: Best checkpoints. We performed MoCov2 Pretext task training on Six datasets for nine epoch and stopped due to a small validation loss change compared with previous epochs. The intention behind the ME2 experiment was to increase the amount of data used in the Pretext task and compare the performance of the resulted Fine-tuned model with the result of Pipeline 2 in the ME1 experiment. According to Fig. 4.3b, the Fine-tuned model reached the best value of validation loss of 0 .4038 on 10^{th} epoch during Downstream task training in the ME2 experiment. This value of validation loss is lower than the best loss value of Fine-Tuned model in the ME1 experiment.

4.4.2 ME3 and ME4 experiments

The purpose of ME3 and ME4 experiments was to see how the MoCov2 Pretext task impacts model performance on the Downstream task if Labeled



(a) ME1 experiment.



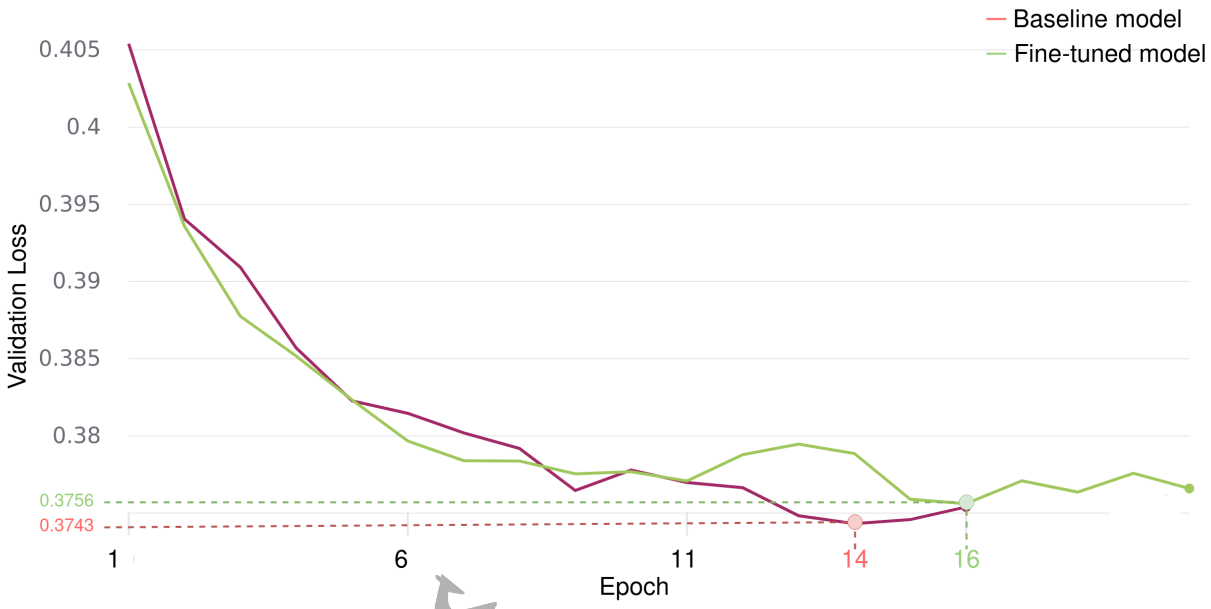
(b) ME1 and ME2 experiments. The Baseline model is same for both of them.

Figure 4.3: The change of validation loss over training epochs during Downstream task training of ME1 and ME2 experiments. For all figures, Baseline model is the result of Pipeline 1, Fine-tuned model – of Pipeline 2. Dotted lines indicate the lowest validation loss of each run and corresponding epoch of training.

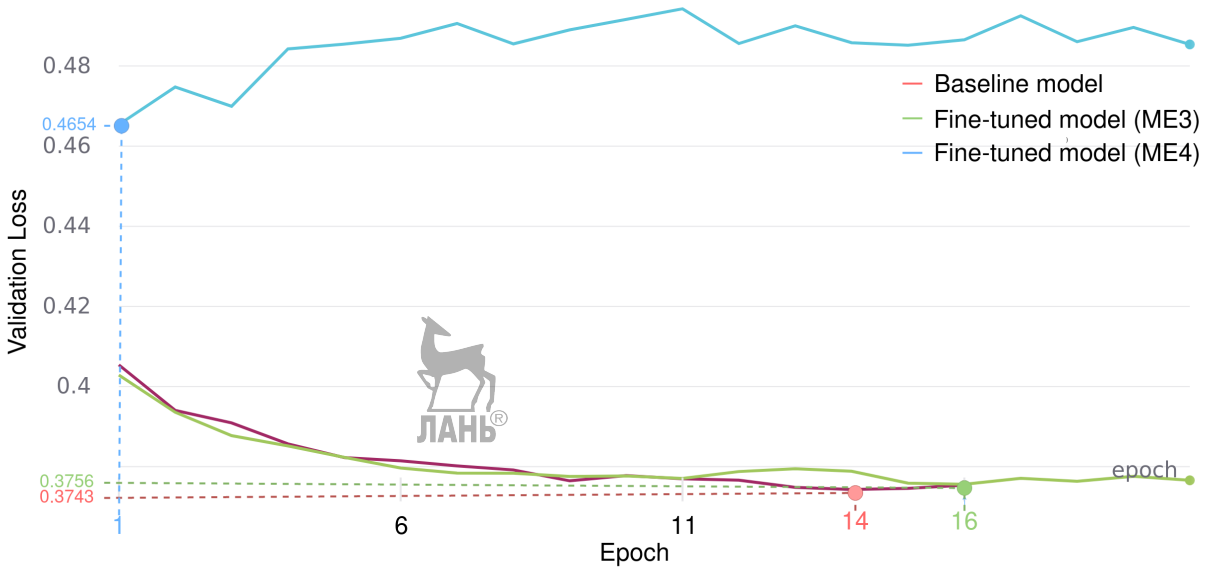
datasets have a small number of examples. In Section 3.4 we introduced the CheXpert-Rare dataset, which is the subset of full CheXpert containing six underrepresented classes. Using this subset as the Labeled dataset, we analysed MoCov2 compatibility to deal with the lack of data for the Downstream task.

ME3: Best checkpoints. In this experiment, we reused the model pre-trained with MoCov2 Pretext task on full CheXpert from the ME1 experiment. According to 4.4a, the best value of validation loss for the Baseline model appeared on 14^{th} epoch and was equal to 0.3743. At the same time, the Fine-tuned model reached the best validation loss of 0.3756 on 16^{th} epoch during Downstream task training.

ME4: Best checkpoints. In this experiment, we reused the model pre-trained with MoCov2 Pretext task on Six Datasets from the ME2 experiment. Fig. 4.4b shows that in the ME4 experiment, Fine-tuned model reached the lowest value 0.4654 of validation loss on the 1^{st} epoch of Downstream task training. This value is higher than 0.3756, which is reached by Fune-tuned model in ME3 experiment. All subsequent epochs resulted in a higher validation loss value.



(a) ME3 experiment.



(b) ME3 and ME4 experiments. The Baseline model is same for both of them.

Figure 4.4: The change of validation loss over training epochs during Downstream task training of ME3 and ME4 experiments. For all figures, Baseline model is the result of Pipeline 1, Fine-tuned model – of Pipeline 2. Dotted lines indicate the lowest validation loss of each run and corresponding epoch of training.

4.5 SimCLR experiment

The SimCLR experiment was the copy of the ME1 run with only one difference. This difference may be seen in Fig. 4.5, where instead of the MoCov2 Pretext task, we used SimCLR Pretext task for model pretraining in Pipeline 2. We conducted this experiment to see how the SimCLR method changes the model performance on the Downstream task.

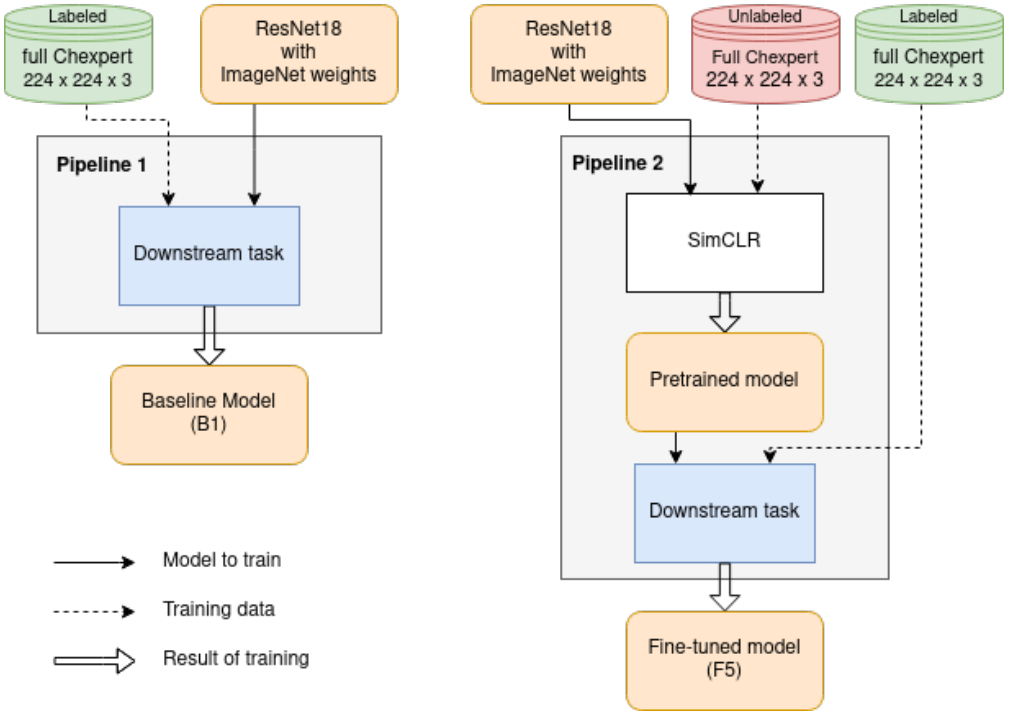


Figure 4.5: The pipelines of SimCLR experiment. It differs from ME1 experiment only in the choice of Pretext Task.

4.5.1 Best checkpoint.

Having Pipeline 1 the same as in the ME1 experiment, we did not create a new Baseline model but compared the new Fune-tuned model with Baseline from ME1. From Fig. 4.6 it is noticeable that Fine-tuned model reached the lowest value of validation loss on 16^{th} epoch. Although the lowest value of

validation loss was 2.011, it was still higher than the largest value of this loss during Baseline model training.

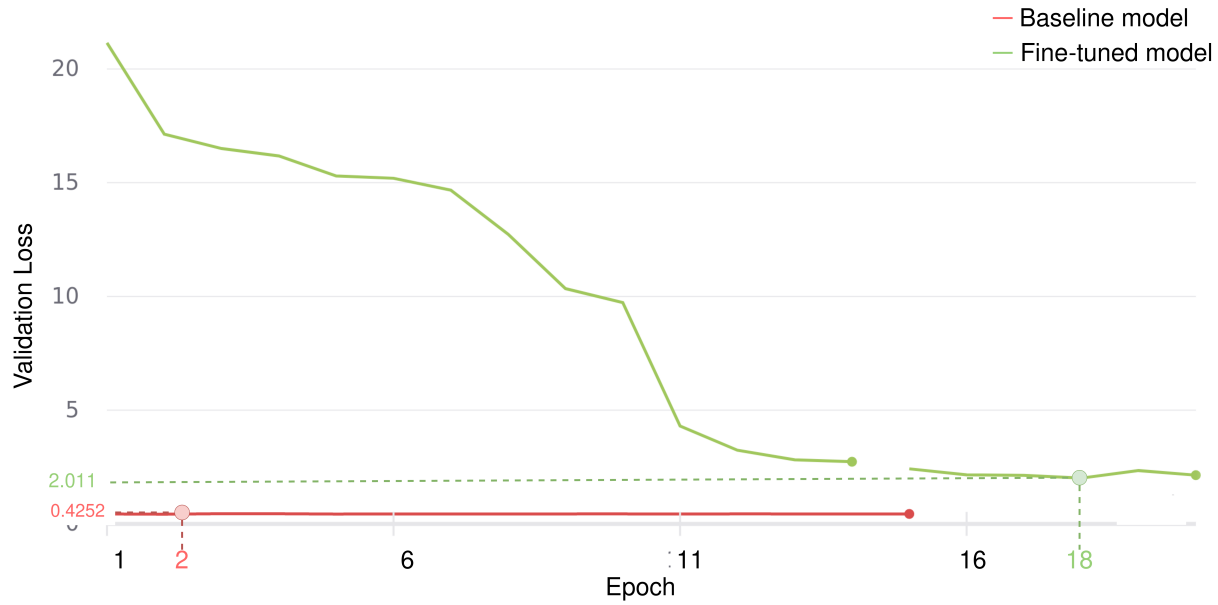


Figure 4.6: The change of validation loss over training epochs during Downstream task training of SimCLR experiment. On this figure, Baseline model is the result of Pipeline 1, Fine-tuned model – of Pipeline 2. Dotted lines indicate the lowest validation loss of each run and corresponding epoch of training.

4.5.2 Conclusion

In this chapter, we described all technical details of our experiments implementation and explained how we chose best models for evaluation. In Chapter 5 we interpret and discuss the outcomes of AUROC metric values, the limitations of our study, and suggest several the paths of future work in this topic.

Chapter 5

Evaluation and Discussion

5.1 Navigation

In this Chapter we evaluate our models and interpret observed outcomes. We also formulate the reasons behind such results and indicate the limitations of our methods.

5.2 Evaluation and interpretation

We evaluated our models using the AUROC metric. Moreover, we have computed 95% Confidence Intervals for it to understand the variability of each estimate. To simplify the data analysis we visualised Confidence Intervals from Tables III, IV and V . In the following sections, we extract important information from each visualization and suggest why such observations occurred. Generally, we are looking for significant shifts of confidence intervals, which may indicate the model improvement or degradation. By significantly shifted interval, we mean such an interval that does not overlap with the centre of the Baseline interval.

Table III: The AUROC metric values with 95% CI retrieved by bootstrap testing of Baseline and Fine-tuned models from ME1 and ME2 experiments. B1 means Baseline model, F1 is the Fine-tuned model from ME1 experiment, F2 is the Fine-tuned model from ME2 experiment. Total value is the Weighted average of AUROC values within all classes.

Class Name	B1	F1	F2
Support Devices	0.656 ± 0.073	0.744 ± 0.065	$\mathbf{0.762 \pm 0.064}$
Lung Opacity	0.794 ± 0.067	0.859 ± 0.051	$\mathbf{0.876 \pm 0.049}$
Pleural Effusion	0.849 ± 0.051	0.889 ± 0.042	0.898 ± 0.040
Atelectasis	0.771 ± 0.067	$\mathbf{0.813 \pm 0.058}$	$\mathbf{0.812 \pm 0.060}$
Edema	0.748 ± 0.078	$\mathbf{0.860 \pm 0.058}$	$\mathbf{0.860 \pm 0.056}$
Consolidation	0.741 ± 0.084	0.767 ± 0.085	0.845 ± 0.063
Cardiomegaly	0.535 ± 0.086	$\mathbf{0.711 \pm 0.076}$	$\mathbf{0.719 \pm 0.080}$
Pneumonia	0.586 ± 0.239	0.671 ± 0.185	$\mathbf{0.809 \pm 0.080}$
Enlarged Cardiomediastinum	0.314 ± 0.071	0.456 ± 0.077	$\mathbf{0.552 \pm 0.080}$
Pneumothorax	0.555 ± 0.249	0.592 ± 0.247	0.666 ± 0.209
No Finding	0.843 ± 0.069	0.891 ± 0.048	0.904 ± 0.046
Lung Lesion	0.157 ± 0.047	$\mathbf{0.405 \pm 0.062}$	0.105 ± 0.045
Fracture	–	–	–
Pleural Other	0.112 ± 0.038	0.854 ± 0.048	$\mathbf{0.918 \pm 0.037}$
Total	0.660 ± 0.032	0.746 ± 0.030	$\mathbf{0.780 \pm 0.025}$

5.2.1 ME1 and ME2 experiments

AUROC metric values We performed non-parametric bootstrap testing of Baseline and Fine-tuned models to compute 95% Confidence Intervals, presented in Table III. The AUROC metric increased for Fine-tuned models comparing with Baseline values. Moreover, Fine-tuned model from the ME2 experiment showed higher AUROC metrics than Fine-tuned model from the ME1 experiment.

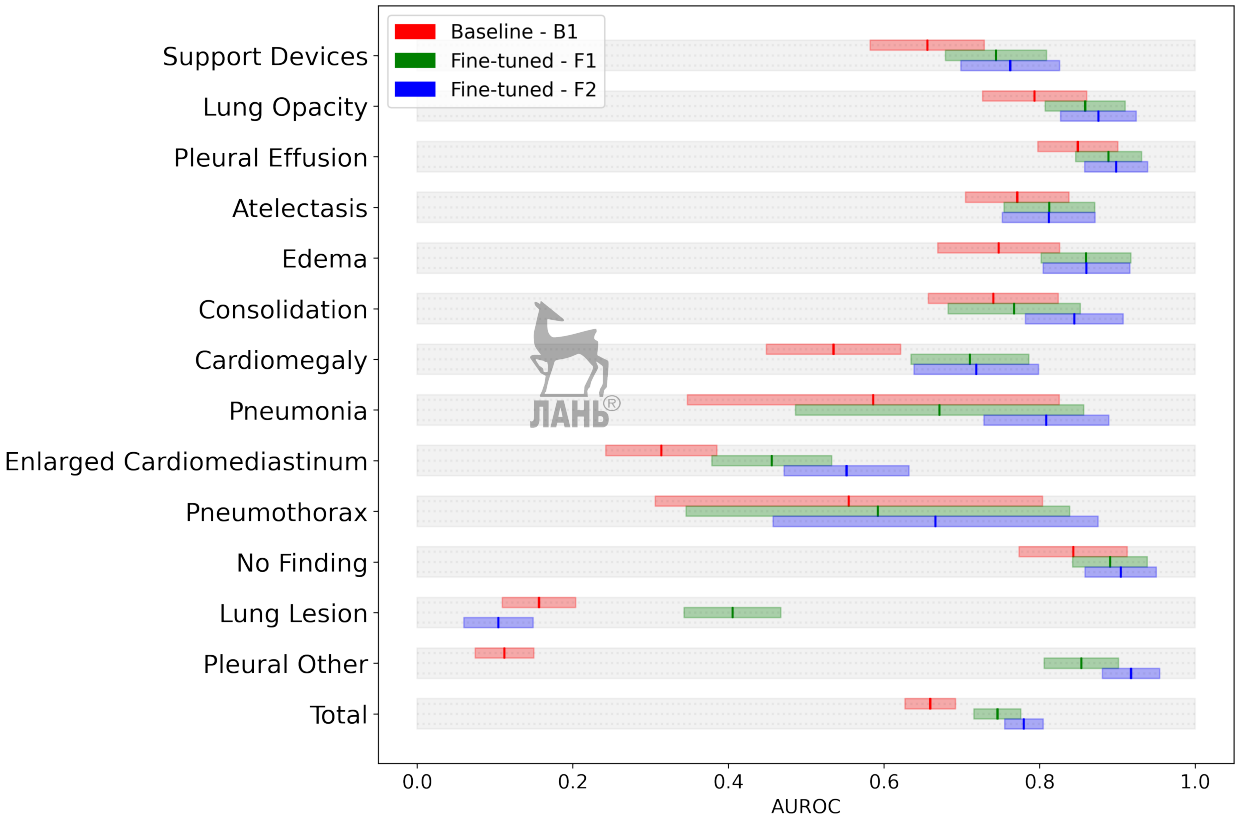


Figure 5.1: The visualisation of 95% Confidence Intervals of AUROC metric for Baseline and Fine-tuned models in ME1 and ME2 experiments.

Interpretation From Fig. 5.1 that depicts Confidence Intervals from Table III it may be seen that the AUROC intervals of Fine-tuned models are shifted towards 1 comparing with outcomes from Baseline B1 model.

Fine-tuned model from the ME1 experiment (F1) significantly improved the classification of nine classes: Support Devices, Lung Opacity, Pleural Effusion, Edema, Cardiomegaly, Enlarged Cardiomediastinum, No Finding, Lung Lesion and Pleural Other. Although the shift is not significant for other classes according to our definition, they still shifted towards 1. Such observation indicates that the model pretraining on the Pretext task does improve its performance on the Downstream task.

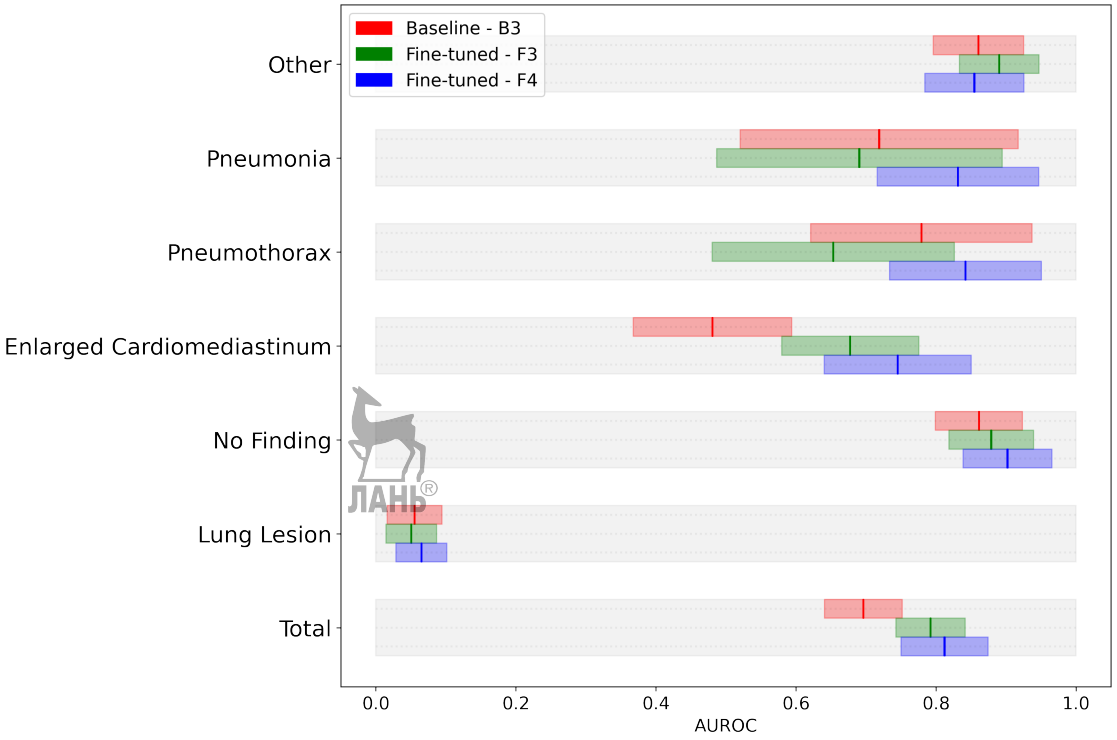
AUROC intervals for another Fine-tuned model, F2 from the ME2 exper-

Table IV: The AUROC metric values with 95% CI retrieved by bootstrap testing of Baseline and Fine-tuned models from ME3 and ME4 experiments. B3 means Baseline model, F3 is the Fine-tuned model from ME3 experiment, F4 is the Fine-tuned model from ME4 experiment. Total value is the Weighted average of AUROC values within all classes.

Class Name	B3	F3	F4
Other	0.861 ± 0.064	0.890 ± 0.057	0.855 ± 0.071
Pneumonia	0.719 ± 0.199	0.691 ± 0.204	0.831 ± 0.115
Pneumothorax	0.779 ± 0.158	0.653 ± 0.173	0.842 ± 0.109
Enlarged Cardiomediastinum	0.481 ± 0.113	0.678 ± 0.098	0.745 ± 0.105
No Finding	0.861 ± 0.062	0.879 ± 0.060	0.902 ± 0.063
Lung Lesion	0.055 ± 0.039	0.051 ± 0.036	0.065 ± 0.036
Fracture	–	–	–
Total	0.696 ± 0.056	0.792 ± 0.049	0.812 ± 0.062

iment, shifted towards 1 as well, except for the Lung Lesion class. This model was significantly better than the Baseline in classifying ten classes: Support Devices, Lung Opacity, Pleural Effusion, Edema, Consolidation, Cardiomegaly, Pneumonia, Enlarged Cardiomediastinum, No Finding and Pleural Other. This fact is another evidence for pretraining effectiveness.

In the case of comparing both F1 and F2 models with Baseline, we can observe that for most classes, the confidence intervals are significantly sifted towards 1. However, comparing F1 to F2, the difference of AUROC intervals is significant only for four classes. Therefore, we do not have enough evidence that for full CheXpert, the effectiveness of the Pretext task changes when the size of the Unlabeled dataset increases.



H

Figure 5.2: The visualisation of 95% Confidence Intervals of AUROC metric for Baseline and Fine-tuned models in ME3 and ME4 experiments.

5.2.2 ME3 and ME4 experiments

AUROC metric values We performed non-parametric bootstrap testing of Baseline and Fine-tuned models to compute 95% Confidence Intervals, presented in Table IV. The resulted AUROC values for the Baseline model is close to Fine-tuned model metrics, except for Enlarged Cardiomediastinum. For this class, Fine-tuned models produce a higher AUROC metric than the Baseline model.

Interpretation Fig. 5.2 visualises the Confidence Intervals from Table IV and shows that for CheXpert-Rare classification it is hard to say if the MoCov2 pretraining was effective. Comparing with Baseline (B3) model, Fine-tuned model from the ME3 experiment (F3) shows significant improvement of the

AUROC metric only for the Enlarged Cardiomediastinum class. The same observation is relevant for Fine-tuned model from the ME4 experiment (F4). Therefore, in our task of rare deceases classification, we have not observed significant evidence that the Pretext task affects model performance on the Downstream task. However, the AUROC intervals of the F4 model are almost two times smaller for Pneumonia and Pneumothorax classes. This fact may indicate the positive impact of Unlabeled dataset expansion on Pretext task effectiveness but require future investigation.

5.2.3 SimCLR experiment interpretation

AUROC metric values We performed non-parametric bootstrap testing of Baseline and Fine-tuned models to compute 95% Confidence Intervals, presented in Table V. This table shows that AUROC metrics of Fine-tuned model are lower than AUROC of Baseline models for most classes.

Interpretation For the SimCLR experiment, the picture is more definite than it was in MoCov2 experiments. From Fig. 5.3 that depicts Confidence Intervals from Table V it can be seen that AUROC metric for Fine-tuned model (F5) is significantly worse than for Baseline model. Although it is true for nine classes: Support Devices, Lung Opacity, Pleural Effusion, Atelectasis, Edema, Consolidation, Cardiomegaly, Pneumonia and No Finding, there are three classes in which the model performance improved significantly: Pneumothorax, Lung Lesion and Pleural Other. However, these improved classes have a low number of positive examples in the test set. Therefore the non-randomness of such outcomes is doubtful.

This experiment showed that not all Pretext tasks are practical and im-

Table V: The AUROC metric values with 95% CI retrieved by bootstrap testing of Baseline and Fine-tuned models from SimCLR experiment. Total value is the Weighted average of AUROC values within all classes.

Class Name	B1	F5
Support Devices	0.656 ± 0.073	0.494 ± 0.077
Lung Opacity	0.794 ± 0.067	0.371 ± 0.074
Pleural Effusion	0.849 ± 0.051	0.347 ± 0.079
Atelectasis	0.771 ± 0.067	0.406 ± 0.083
Edema	0.748 ± 0.078	0.198 ± 0.066
Consolidation	0.741 ± 0.084	0.598 ± 0.108
Cardiomegaly	0.535 ± 0.086	0.339 ± 0.084
Pneumonia	0.586 ± 0.239	0.353 ± 0.258
Enlarged Cardiomediastinum	0.314 ± 0.071	0.269 ± 0.065
Pneumothorax	0.555 ± 0.249	0.663 ± 0.158
No Finding	0.843 ± 0.069	0.570 ± 0.095
Lung Lesion	0.157 ± 0.047	0.920 ± 0.035
Fracture	–	–
Pleural Other	0.112 ± 0.038	0.801 ± 0.055
Total	0.660 ± 0.032	0.380 ± 0.036

prove model performance on the Downstream task. We suggest that the set of such intense image augmentation used in the SimCLR Pretext task is unsuitable for CXR data. In our opinion, the incompatibility of image transformations with data type was the primary reason for observed model degradation.

5.3 Limitations

Our study has several essential limitations that may have influenced achieved results and formulated interpretations.

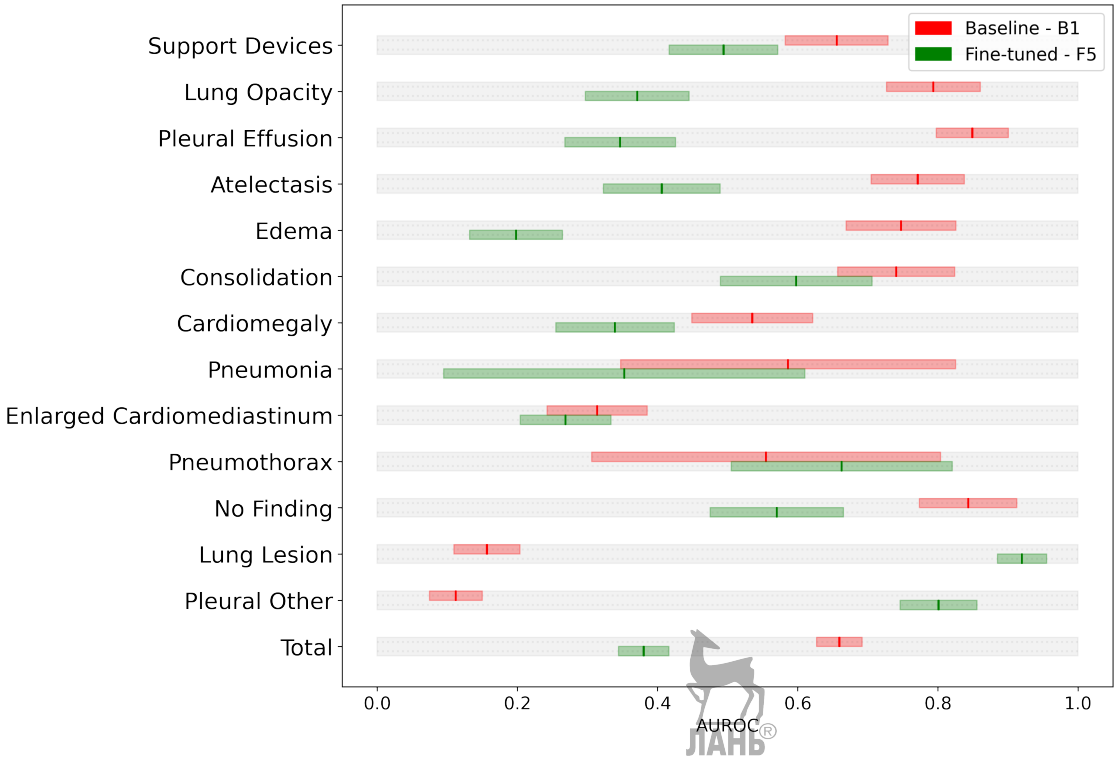


Figure 5.3: The visualisation of 95% Confidence Intervals of AUROC metric for Baseline and Fine-tuned models in SimCLR experiment.

The first limitation is connected to the time of model training. We did not train models on the Downstream task for more than 20 epochs, which means that we may have not fully used the ResNet18 abilities to train. Therefore, we may have compared not actual model performances.

The second limitation refers to the data storage throughput. We were limited to a maximum of 32 images per batch for all training processes. This batch size may have been not suitable for the SimCLR Pretext task. However, it requires future investigation to find the proof.

The last limitation we have found in the CheXpert test set. This set contains only 202 images with an unbalanced distribution of classes. Although it is usually not critical for multi-label classification evaluation, we could not objectively evaluate the classification accuracy for classes with less than ten or even less than five positive examples in the test set.

We suggest ways to eliminate listed limitation and improve the significance of collected results in the following section.

5.4 Future Work

There are several possible directions of future research in the area of Self-Supervised learning for CXR images classification.

Firstly, in our opinion, it is crucial to analyze how different image transformations affect the MoCov2 pretraining. This direction implies the comparison analysis and the creation of rating with the most effective combinations suitable for CXR images.

Secondly, the idea of Unlabeled dataset expansion may be further studied with different distributions of findings and data origin. It is necessary to understand what properties are essential for Unlabeled data. The same direst ion may include creating a universal CXR images normalization method, which may be helpful to unite datasets from different institutes.

For the last direction, we suggest applying other Contrastive and Generative Self-Supervised algorithms to CXR images and compare the outcomes with our findings.

Chapter 6

Conclusion

The lack of labeled data is a crucial problem for the majority of machine learning tasks. This problem is especially relevant in the case of medical image analysis. To collect reliable labels for medical examples, researchers invite several medical doctors for data review. This process is costly in time. Therefore most of the medical datasets are limited in size.

Self-Supervised methods introduce the idea of training machine learning models with unlabeled datasets. This idea is a promising method for medical models pretraining, as medical institutes have large volumes of suitable unlabeled images. Still, there was a limited study on the effectiveness of Self-Supervised pretraining methods applied to medical data. The lack of studies on this topic is especially relevant for the case of chest X-ray's analysis. Detection of respiratory diseases is a prevalent task in medical practice, therefore in this study, we decided to answer the following research question: «Does Self-Supervised pretraining improves chest X-ray's classification?». Moreover, we were interested in the situation when the amount of training data is limited. Therefore, we have defined a list of rare diseases and focused on answering the

same question for a limited set of labeled data. Finally, we expanded the pretraining dataset with relevant images from different sources to study if more data in Self-Supervised pretraining causes more precise model prediction in a subsequent classification task.

To perform the experiments we used CheXpert dataset [21] for classification as well as for Self-Supervised pretraining. Additionally, we used images from ChestX-ray14 [19], Chest X-Ray Pneumonia [27], GB7-FLG, TBX11K [28], and Vinbigdata [29] datasets to expand pretraining dataset in some of experiments. Fig. 3.4. presents the general pipeline for each test, whereas Fig. 4.2 and Fig. 4.5 specifies the pipelines of experiments for concrete Self-Supervised methods used. These methods are MoCov2 and SimCLR accordingly.

To understand the significance of our results, we applied non-parametric bootstrap testing to all our models. We performed 500 test runs on random samples from the test dataset during one model testing and used the collected distribution of AUROC metric to calculate 95% Confidence Interval for evaluation estimate. Obtained results are listed in Table III, Table IV and Table V.

From these data visualisation presented in Fig. 5.1, Fig. 5.2 and Fig. 5.3 we made a conclusion that MoCov2 pretraining task improves the accuracy of CheXpert images classification comparing with our Baseline model. In contrast, we have not observed significant evidence that model performance in classification improves after pretraining dataset expansion. Additionally, we have not received significant evidence that the model with pretraining significantly differs from the model without pretraining if trained only on our subset of rare classes from CheXpert. Finally, the experiment with SimCLR pretraining indicated that if strong augmentations are applied during this stage, the model performs

significantly worse during classification than a model without pretraining.

Based on a statistical analysis of retrieved AUROC values, we presented our interpretation of observed outcomes. However, there is several limitations that may have affected our results. They refer to limited model training, data storage throughput and available test samples.

At the end of our discussion, we suggested several directions for future research in the Self-Supervised learning application for chest X-ray classification. These digestions may be shortly described with the following statements:

- Compare different image transformations in the context of MoCov2 model pretraining to identify the most suitable augmentations for chest X-ray data.

- Conduct a more thoughtful analysis of how dataset expansion in the pretraining stage affects subsequent model performance on the task of medical image classification. This analysis may imply creating a universal normalisation procedure for chest X-ray images, which will be useful to eliminate bias shift between datasets from different institutes.

- Study how other Self-Supervised algorithms affect chest X-ray's classification to identify the state of the art method for the data of this type.

With suggested improvements, we believe that the Self-Supervised methods will be further applied to improve medical image analysis and hope that the results of our study will be used for future research in this field.

Bibliography cited

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[2] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 01, pp. 1–1, May 5555, ISSN: 1939-3539. DOI: 10.1109/TPAMI.2020.2992393.

[3] X. Liu, F. Zhang, Z. Hou, Z. Wang, L. Mian, J. Zhang, and J. Tang, *Self-supervised learning: Generative or contrastive*, 2020. arXiv: 2006.08218 [cs.LG].

[4] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ser. ICCV '15, USA: IEEE Computer Society, 2015, pp. 1422–1430, ISBN: 9781467383912. DOI: 10.1109/ICCV.2015.167. [Online]. Available: https://doi.org/10.1109/ICCV.2015.167.

[5] D. Kim, D. Cho, D. Yoo, and I. S. Kweon, "Learning image representations by completing damaged jigsaw puzzles," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 793–802. DOI: 10.1109/WACV.2018.00092.

[6] N. Komodakis and S. Gidaris, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations (ICLR)*, Vancouver, Canada, Apr. 2018. [Online]. Available: https://hal-enpc.archives-ouvertes.fr/hal-01832768.

[7] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., Cham: Springer International Publishing, 2018, pp. 139–156, ISBN: 978-3-030-01264-9.

[8] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3733–3742. DOI: 10.1109/CVPR.2018.00393.

[9] D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," in *ICLR 2019*, ICLR, Apr. 2019. [Online]. Available: https://www.microsoft.com/en-us/research/publication/learning-deep-representations-by-mutual-information-estimation-and-maximization/.

[10] A. V. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *Proceedings of The 33rd International Conference on*

Machine Learning, M. F. Balcan and K. Q. Weinberger, Eds., ser. Proceedings of Machine Learning Research, vol. 48, New York, New York, USA: PMLR, Jun. 2016, pp. 1747–1756. [Online]. Available: http://proceedings.mlr.press/v48/oord16.html.

[11] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17, Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6309–6318, ISBN: 9781510860964.

[12] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *European Conference on Computer Vision (ECCV)*, 2016.

[13] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros, "Context encoders: Feature learning by inpainting," 2016.

[14] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 105–114. DOI: 10.1109/CVPR.2017.19.

[15] N. Tajbakhsh, Y. Hu, J. Cao, X. Yan, Y. Xiao, Y. Lu, J. Liang, D. Terzopoulos, and X. Ding, "Surrogate supervision for medical image analysis: Effective deep learning from limited quantities of labeled data," English (US), in *ISBI 2019 - 2019 IEEE International Symposium on Biomedical Imaging*, ser. Proceedings - International Symposium on Biomedical Imaging, Publisher Copyright: © 2019 IEEE. Copyright: Copyright

2019 Elsevier B.V., All rights reserved.; 16th IEEE International Symposium on Biomedical Imaging, ISBI 2019 ; Conference date: 08-04-2019 Through 11-04-2019, IEEE Computer Society, Apr. 2019, pp. 1251–1255. DOI: 10.1109/ISBI.2019.8759553.

[16] N. Tajbakhsh, L. Jeyaseelan, Q. Li, J. N. Chiang, Z. Wu, and X. Ding, "Embracing imperfect datasets: A review of deep learning solutions for medical image segmentation," *Medical Image Analysis*, vol. 63, p. 101 693, 2020, ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2020.101693. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S136184152030058X.

[17] L. Chen, P. Bentley, K. Mori, K. Misawa, M. Fujiwara, and D. Rueckert, "Self-supervised learning for medical image analysis using image context restoration," *Medical Image Analysis*, vol. 58, p. 101 539, 2019, ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2019.101539. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1361841518304699.

[18] A. Jamaludin, T. Kadir, and A. Zisserman, "Self-supervised learning for spinal mris," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, M. J. Cardoso, T. Arbel, G. Carneiro, T. Syeda-Mahmood, J. M. R. Tavares, M. Moradi, A. Bradley, H. Greenspan, J. P. Papa, A. Madabhushi, J. C. Nascimento, J. S. Cardoso, V. Belagiannis, and Z. Lu, Eds., Cham: Springer International Publishing, 2017, pp. 294–302, ISBN: 978-3-319-67558-9.

[19] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on

weakly-supervised classification and localization of common thorax diseases," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017. DOI: 10.1109/cvpr.2017.369. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2017.369.

[20] A. E. W. Johnson, T. J. Pollard, S. J. Berkowitz, N. R. Greenbaum, M. P. Lungren, C.-y. Deng, R. G. Mark, and S. Horng, "MIMIC-CXR: A large publicly available database of labeled chest radiographs," *CoRR*, vol. abs/1901.07042, 2019. arXiv: 1901.07042. [Online]. Available: http://arxiv.org/abs/1901.07042.

[21] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. L. Ball, K. S. Shpanskaya, J. Seekins, D. A. Mong, S. S. Halabi, J. K. Sandberg, R. Jones, D. B. Larson, C. P. Langlotz, B. N. Patel, M. P. Lungren, and A. Y. Ng, "Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison," *CoRR*, vol. abs/1901.07031, 2019. arXiv: 1901.07031. [Online]. Available: http://arxiv.org/abs/1901.07031.

[22] K. K. Bressem, L. Adams, C. Erxleben, B. Hamm, S. Niehues, and J. Vahldiek, *Comparing different deep learning architectures for classification of chest radiographs*, 2020. DOI: 10.1038/s41598-020-70479-z. [Online]. Available: https://arxiv.org/abs/2002.08991.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1409.1556.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, ISSN: 0001-0782. DOI: 10.1145/3065386. [Online]. Available: https://doi.org/10.1145/3065386.

[26] M. Lenga, H. Schulz, and A. Saalbach, "Continual learning for domain adaptation in chest x-ray classification," in *MIDL*, 2020.

[27] D. S. Kermany, M. Goldbaum, W. Cai, C. C. Valentim, H. Liang, S. L. Baxter, A. McKeown, G. Yang, X. Wu, F. Yan, J. Dong, M. K. Prasadha, J. Pei, M. Y. Ting, J. Zhu, C. Li, S. Hewett, J. Dong, I. Ziyar, A. Shi, R. Zhang, L. Zheng, R. Hou, W. Shi, X. Fu, Y. Duan, V. A. Huu, C. Wen, E. D. Zhang, C. L. Zhang, O. Li, X. Wang, M. A. Singer, X. Sun, J. Xu, A. Tafreshi, M. A. Lewis, H. Xia, and K. Zhang, "Identifying medical diagnoses and treatable diseases by image-based deep learning," *Cell*, vol. 172, no. 5, 1122–1131.e9, 2018, ISSN: 0092-8674. DOI: https://doi.org/10.1016/j.cell.2018.02.010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0092867418301545.

[28] Y. Liu, Y.-H. Wu, Y. Ban, H. Wang, and M.-M. Cheng, "Rethinking computer-aided tuberculosis diagnosis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2646–2655.

[29] H. Q. Nguyen, K. Lam, L. T. Le, H. H. Pham, D. Q. Tran, D. B. Nguyen, D. D. Le, C. M. Pham, H. T. T. Tong, D. H. Dinh, C. D. Do, L. T. Doan,

C. N. Nguyen, B. T. Nguyen, Q. V. Nguyen, A. D. Hoang, H. N. Phan, A. T. Nguyen, P. H. Ho, D. T. Ngo, N. T. Nguyen, N. T. Nguyen, M. Dao, and V. Vu, *Vindr-cxr: An open dataset of chest x-rays with radiologist's annotations*, 2020. arXiv: 2012.15029 [eess.IV].

[30] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9726–9735. DOI: 10.1109/CVPR42600.2020.00975.

[31] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference on Machine Learning*, H. D. III and A. Singh, Eds., ser. Proceedings of Machine Learning Research, vol. 119, PMLR, Jul. 2020, pp. 1597–1607. [Online]. Available: http://proceedings.mlr.press/v119/chen20j.html.

[32] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, 2021, ISSN: 2227-7080. DOI: 10.3390/technologies9010002. [Online]. Available: https://www.mdpi.com/2227-7080/9/1/2.

[33] H. Sowrirajan, J. Yang, A. Y. Ng, and P. Rajpurkar, *Moco-cxr: Moco pretraining improves representation and transferability of chest x-ray models*, 2021. arXiv: 2010.05352 [cs.CV].

[34] S. Azizi, B. Mustafa, F. Ryan, Z. Beaver, J. Freyberg, J. Deaton, A. Loh, A. Karthikesalingam, S. Kornblith, T. Chen, V. Natarajan, and M. Norouzi, *Big self-supervised models advance medical image classification*, 2021. arXiv: 2101.05224 [eess.IV].

[35] e. a. Falcon WA, "Pytorch lightning," *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, vol. 3, 2019.

[36] W. Falcon and K. Cho, "A framework for contrastive self-supervised learning and designing a new approach," *arXiv preprint arXiv:2009.00104*, 2020.