Semi-Supervised Transfer Learning for medical images as an alternative to ImageNet Transfer Learning

Prepared by:
Xolisani Nkwentsha

Prepared for:
Professor Fred Nicolls
Department of Electrical Engineering
University of Cape Town

This report is submitted in fulfillment of the requirements of a
Master of Science in Engineering in Electrical Engineering

June 10, 2022

Declaration

I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report/project from the work(s) of other people has been attributed, and has been cited and referenced.
This report/project is my own work.
I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.

Signature: .....
Xolisani Nkwentsha, NKWXOL003
June 10, 2022

Acknowledgements

I would like to acknowledge Professor Fred Nicolls, Anicet Hounkanrin, Emmanuel Ali, and the rest of the team in the Digital Image Processing Lab and thank them for their support and guidance.

I would also like to thank all my family and friends for their love and support, which made this thesis possible.

Abstract

One of the main disadvantages of supervised transfer learning is that it necessarily requires a large amount of expensive manually labelled training data. Consequently, even in medical imaging, transfer learning from natural image datasets (such as ImageNet) has become the norm. However, this approach has been shown to be ineffective due to the significant differences between medical images and natural images. Developing a large-scale medical imaging dataset for transfer learning would be too expensive, therefore the possibility of using large amounts of unlabelled data for feature learning is very attractive.

In this work, we propose a semi-supervised transfer learning method for training deep learning models for medical imaging. The main idea behind the proposed method is to leverage unlabelled medical image datasets to improve accuracy for the target task by transferring feature maps learned from an unsupervised task to the supervised target task. We leverage unlabelled data by transferring weights/kernels and representations learned by an autoencoder (specifically the encoder part) during a reconstruction task to a classification task.

We show the applicability of features learned by the autoencoder from the collection of unlabelled x-ray images to a pneumonia classification problem. Our proposed method improves the baseline performance by 4.167% in accuracy and the precision, recall and F1 score by 4%. We also demonstrate that increasing the size of the unlabelled dataset used to train the autoencoder improves the performance on the target task. This increase in the size of the dataset resulted in an overall 5.288% accuracy increase from the baseline. We also compare our method with ImageNet models on the target dataset. For the standard ImageNet architectures, we evaluate ResNet50 and Inception-v3, which have both been used extensively in medical deep learning applications. Our proposed method outperforms both standard ImageNet models on the target task.

These results demonstrate that learning features from unlabelled medical images for transfer learning for medical imaging tasks is more effective than transfer learning from natural images, at least for the problem of pneumonia detection.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter provides the reader with an outline of the project's background as well as its overall objectives. It summarizes the project's goals and sets out a plan for the rest of the study.

1.1 Background

Medical facilities are processing vast volumes of medical imaging data as a result of advances in emerging technologies, resulting in an unprecedented growth of medical image archives. Medical imaging has played a vital role in assisting healthcare practitioners over the years by supplying relevant knowledge about anatomical and biological systems, which improves analysis and diagnosis. Rapid advances in medical imaging have created a challenge by generating more data in various modalities. Radiologists must analyze more data while maintaining a high level of consistency and reliability. To help in successful diagnosis, systems that can automatically view, analyze, and categorize medical images are needed[12].

Latest developments in deep learning have provided more robust and consistent tools for the analysis, detection, and classification of patterns in medical images. In several computer vision activities, such as classification [13], object detection [14], and image segmentation [15], convolutional neural networks (CNNs) are providing critical breakthroughs.

The scarce availability of labelled medical imaging data, however, poses a massive barrier to training CNNs due to the expense and required workload on radiologists to be able to label medical images [16]. Therefore, transfer learning from natural image datasets, especially ImageNet (which consists of over 14 million images organized into over 21 thousand subcategories [17]), using standard large ImageNet models (e.g. Inception, VGG, ResNet, Xception etc.) and corresponding pretrained weights has become a norm in deep learning for

medical imaging [16]. However, there are significant differences between medical and natural images (e.g. features, data sizes, and task specifications), and there is little knowledge of the consequences of transferring natural image learning to medical image tasks [18, 19]. Raghu et al. [18] investigate the properties of transfer learning in the context of medical imaging. They show that transfer learning from natural images (i.e. ImageNet model weights and kernels) provides little benefit to the performance of the models, and simple, lightweight models trained from scratch have similar performance to pretrained ImageNet architectures.

1.2 Objectives of this project

The ultimate goal of this research project is to develop a method of learning features from large volumes of unlabelled data for medical image applications.

In this work we propose a semi-supervised transfer learning approach for training deep learning models for medical images, in response to the problem of insufficient labelled medical imaging data. The main idea behind the proposed method leverages unlabelled medical image datasets to improve model accuracy for the target task by transferring feature maps learned from an unsupervised task to the supervised target task. We leverage unlabelled data by transferring weights/kernels and representations learned by an autoencoder (specifically the encoder part) during an unsupervised reconstruction task (from a large collection of unlabelled medical images) to a classification task.

We investigate whether the proposed method improves the performance of the model compared to the baseline (i.e. to train the model from scratch) and state of the art ImageNet models. We also investigate the impact of the size of the unlabelled source dataset, and other characteristics (like whether the unlabelled medical dataset is broad and diverse, homogeneous, or exactly the same as the target dataset) on the performance of the proposed method. The primary objective is divided into a number of sub-objectives:

- Develop and train a small CNN model to perform the classification on a target dataset as the baseline.
- Develop an auto-encoder for an unsupervised reconstruction task and train the autoencoder for reconstruction on a source dataset.
- Transfer kernels/weights learned from a reconstruction autoencoder to the small CNN for classification on the target dataset.

- Evaluate our proposed method's performance against the baseline and state-of-the-art ImageNet models for classification on the target dataset.

1.3 Contributions

The contribution of this work includes the following:

- We implement a semi-supervised learning approach that makes use of an autoencoder to learn features from a collection of unlabelled X-ray images.
- We show the applicability of features learned by the autoencoder from the collection of unlabelled X-ray images to a pneumonia classification problem. We compare our results to ImageNet pre-trained architectures and achieve state-of-the-art results.
- We also demonstrate that increasing the size of the unlabelled dataset used to train the autoencoder improves the performance on the target task.

1.4 Motivation for study

One central downside of supervised transfer learning is that a large quantity of expensive manually labelled training data is required. Hence transfer learning from natural image datasets has become a standard even for medical imaging. However, this approach has been shown to be ineffective due to the significant differences between medical images and natural images [18]. Developing a large-scale medical imaging dataset for transfer learning would be too expensive, therefore the prospect of learning features from large volumes of unlabelled data is promising.

It would be significantly easier and less expensive to collect large amounts of unlabelled medical images. Therefore, successfully leveraging unlabelled data to pretrain models for classification and detection could potentially have an impact on medical imaging similar to ImageNet's impact on computer vision in general.

1.5 Scope and limitations

This work focuses on and is limited to convolutional neural networks for image classification. Today, X-ray imaging is the most often used medical imaging modality, since many healthcare institutions have X-ray scanners and store their own image databases. Therefore, this work will be focusing primarily on X-ray images.

Chest radiography, one of the most popular medical imaging assessments in the clinical workflow, requires prompt documentation of results and diagnosis of potential illnesses in the X-ray image. An important step in the radiology workflow is the fast, automated, and reliable detection and classification of illnesses and diseases. Timely and accurate documenting on all X-ray images is necessary but is not always achievable because of the substantial workload of radiologists in healthcare institutions and the lack of expert radiologists in underdeveloped regions.

In this work we focus on the classification of pneumonia from chest X-rays, although the proposed method can be used for any target detection task in medical imaging (X-ray, CT scan etc.). Pneumonia is one of the main reasons for death and hospitalisation in South Africa [20]. HIV-infected people have a significantly high risk of hospitalisation and death due to pneumonia [20]. According to World Health Organisation (WHO) data published in 2018, influenza and pneumonia deaths in South Africa reached 36,810 or 7.54% of total deaths. Chest radiographs are at present the best available method for diagnosing pneumonia, playing a critical role in clinical workflow and epidemiological studies [21]. However, detecting pneumonia in X-rays images can be a complicated task that relies heavily on the availability of experienced radiology experts [22]. The presence or absence of pneumonia in X-ray images is often unclear and can overlap with other illnesses and can resemble many other abnormalities. These inconsistencies cause significant variability among radiologists in the diagnosis of pneumonia.

1.6 Software, computing and datasets

In this section we detail the resources used in this work.

1.6.1 Software

A variety of software tools were used, most notably Python and the deep learning frameworks Tensorflow and Keras. Additional notable software packages used include matplotlib, numpy, pandas, h5py, and openCV.

1.6.2 Computing

Computations were performed using facilities provided by the University of Cape Town's ICTS High Performance Computing team: `http://hpc.uct.ac.za`. The experiments were mainly performed on 4 NVIDIA KeplerK80M GPUs with a RAM of 12GB per GPU.

1.6.3 Datasets

For this project, experiments were done using the Guangzhou Women and Children's Medical Center Pediatric Dataset, the ImageCLEF 2009 dataset, and the RSNA pneumonia detection challenge dataset:

1. Guangzhou Women and Children's Medical Center Pediatric Dataset (GWCMCPD)
   This dataset has 5856 chest X-rays made public by the Guangzhou Women and Children's Medical Center (GWCMC). X-rays images in the dataset are either labelled as normal or pneumonia. The dataset is split into train, test and validation sets, where 5232 images (1349 normal, 3883 pneumonia) are part of the training and validation set, and the remaining 624 X-rays (234 normal, 390 pneumonia) are used for testing.

2. ImageCLEFmed 2009
   The ImageCLEFmed 2009 dataset is a collection of anonymous radiographs selected from the Department of Diagnosis Radiology at the Aachen University of Technology (RWTH), Germany. The dataset is made up of X-rays obtained from patients of different ages and genders, resulting in a highly diverse collection. All images in the dataset are downscaled to fit into a 512×512 bounding box. The dataset is partitioned into two sets: the training set has 12677 images labelled according to the IRMA code and the test set has 1733 images.

3. RSNA pneumonia detection challenge dataset
   Six board-certified radiologists from the Radiological Society of North America (RSNA) and two radiologists from the Society of Thoracic Radiology (STR) re-labelled a total of 25684 chest X-rays from the NIH database into three groups: normal (8525), abnormal with lung opacity (5659) and abnormal without lung opacity (11500). The definition of "pneumonia-like lung opacity" includes findings like pneumonia, infiltration, consolidation, and other lung opacities that radiologists consider as pneumonia-related.

1.7 Plan of development

This report begins with a brief background on fundamental concepts of deep learning with a focus on convolutional neural networks in *Chapter 2*. *Chapter 3* provides a review of popular deep learning based classification approaches focusing on convolutional neural networks, followed by brief discussion about the availability of datasets for medical imaging tasks. *Chapter 4* provides a brief background on the autoencoder and its applications in

medical imaging.

Chapter 5 gives a detailed description of the proposed method, and the architectures, datasets and preprocessing techniques used. By the end of this chapter the reader should be able to replicate the work done in this project. In *Chapter 6* we set out the experimental setup, key experimental results and discussion of the results in the context of the main question of the project.

Finally, in *Chapter 7* we formulate conclusions in order to determine the significance of the results to the main question. We end *Chapter 7* with suggestions for future work.

Chapter 2

Deep learning background

Machine learning is driving countless elements of contemporary society: from internet searches to filtering of content on social media networks to suggestions on websites. It has become more and more present in consumer devices such as smartphones and wearable devices [23]. Machine learning is applied in many industries that include healthcare, retail, security, transportation, and finance. Machine learning techniques are used to find objects in images, perform automatic speech recognition, and recommend adverts based on users' interests and other personal information. Recently, these implementations used a group of approaches called deep learning [23].

Machine learning methods are restricted by their inability to automatically extract features from data in their raw form. Developing a machine learning system requires diligent feature engineering and a large amount of domain expertise to design and create a feature extractor that can transform raw data into a relevant representation or feature vector from which the learning algorithm can detect and recognise patterns in the input data [23].

Representation learning is a collection of techniques that allow machines to automatically find the representations required for the task at hand (classification, detection, etc.) from raw data [23]. Deep learning approaches are methods of learning with several layers of representation, obtained through creating non-linear modules, each translating the representation at each stage (starting from the input) into a representation at a higher, somewhat more abstract representation [23]. Very complex functions can be learned with the sufficient creation of such translations. Higher layers of representation intensify elements of the input data that are more essential for discrimination and remove trivial variations for classification tasks.

For example, an image usually comes in the form of an array of pixel values. In the first layer of the representation, the learned features usually represent the presence of edges at specific locations in the image. By detecting specific arrangements of edges, the second layer usually identifies patterns, irrespective of minor variations in the position of the edge. The third layer will combine patterns from the second layer into larger variations that correspond to recognizable object pieces, and objects can be identified by subsequent layers as combinations of these pieces (see Figure 2.1). The major characteristic of deep learning techniques is that these features and representations are not developed by humans, but rather they are learned from raw data [23].

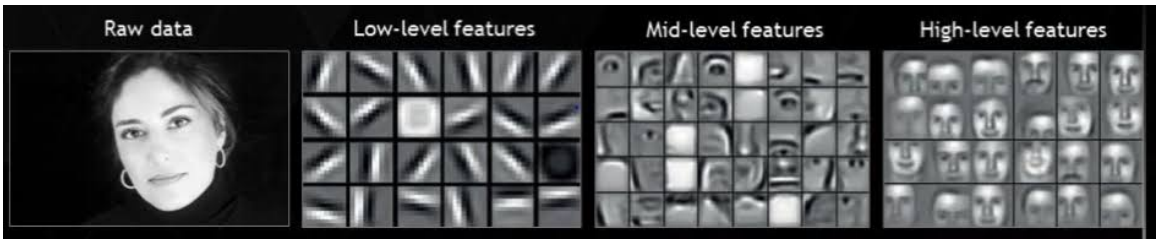

Figure 2.1: Hierarchical representation learning by a convolutional neural network where the low-level layers extract basic features like edges and corners while high-level layers extract more abstract features[1].

In recent years, a significant interest in deep learning has arisen [23]. The most popular architecture among many deep learning techniques is convolutional neural networks (CNN), a type of artificial neural network that has proven to be effective in multiple computer vision tasks like image classification [13], object detection [14], and image segmentation [15]. Extremely impressive results were shared by Krizhevsky et al. [2] on the ImageNet Large Scale Visual Recognition Competition (ILSVRC) in 2012 [3]. Krizhevsky et al. [2] developed AlexNet (see Figure 2.2), which showed the potential of convolutional neural networks in computer vision tasks.

AlexNet is regarded as one of the most important papers in computer vision, having sparked the publication of many papers that used CNNs and GPUs to facilitate deep learning [24]. The paper used a CNN to achieve a top-5 error rate of 15.3%. The next best performance trailed well behind at 26.2% [24]. A shift toward convolutional neural networks has been seen recently. Out of the 47 papers published on exam classification in 2015, 2016, and 2017, 36 use CNNs, 5 are based on autoencoders and 6 on restricted Boltzmann machines [16].

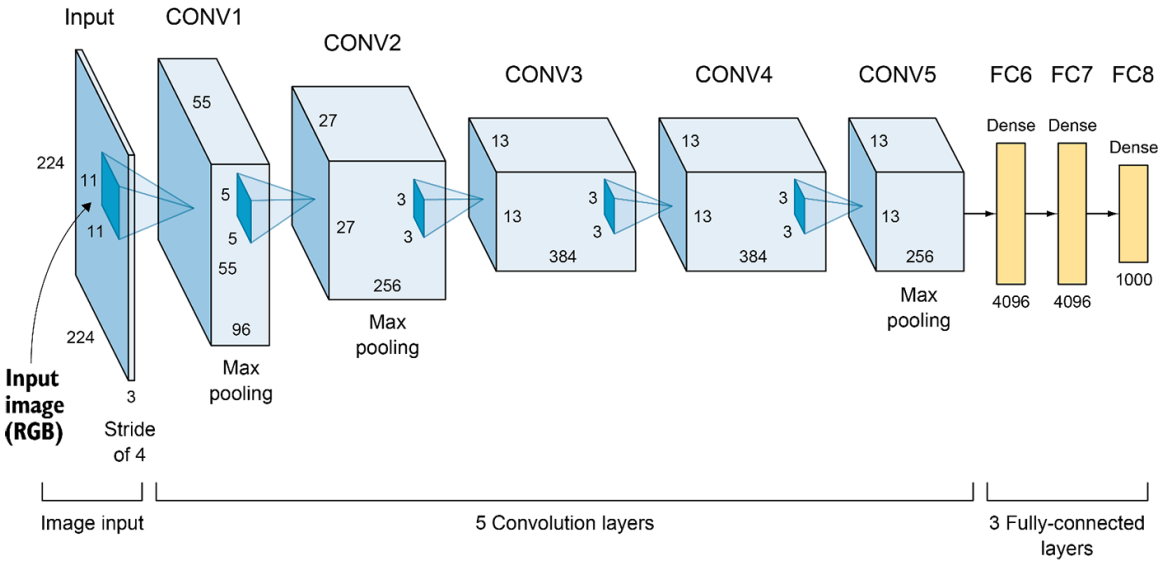

Figure 2.2: AlexNet is an eight-layer architecture with five convolutional layers and three fully-connected layers. This architecture showed the potential of convolutional neural networks in computer vision tasks in 2012 by winning the ILSVRC have an error rate of 15.3% (top-5), the next best architecture achieved an error rate of 26.2% [2].

In this chapter we provide a theoretical background to the basics of convolutional neural networks that is required to have a good understanding of the concepts used in this work. We do not perform a critical literature review in this chapter as it's purpose is to introduce the reader to general concepts in deep learning used in the remainder of the report. We do a more critical literature review in the following chapter.

2.1 Convolutional Neural Networks

A convolutional neural network is a type of deep learning model that can operate on data that comes in the form of multiple arrays, for example an image which has 2D arrays containing pixel intensities (see Figure 2.3) [3]. There are several kinds of data in the form of multiple arrays, for example 1D signals and sequences, 2D images or audio spectrograms, and 3D video or 3D images [23]. CNNs are programmed to learn spatial function hierarchies automatically, from low to high-level features [3]. There are four main concepts behind CNNs that make use of the qualities of natural signals: shared weights, pooling local connections, and the use of multiple layers [23].

CNNs are made up of three different kinds of layers, namely convolution, fully-connected layers, and pooling. The pooling and convolutional layers act as feature extractors, and the

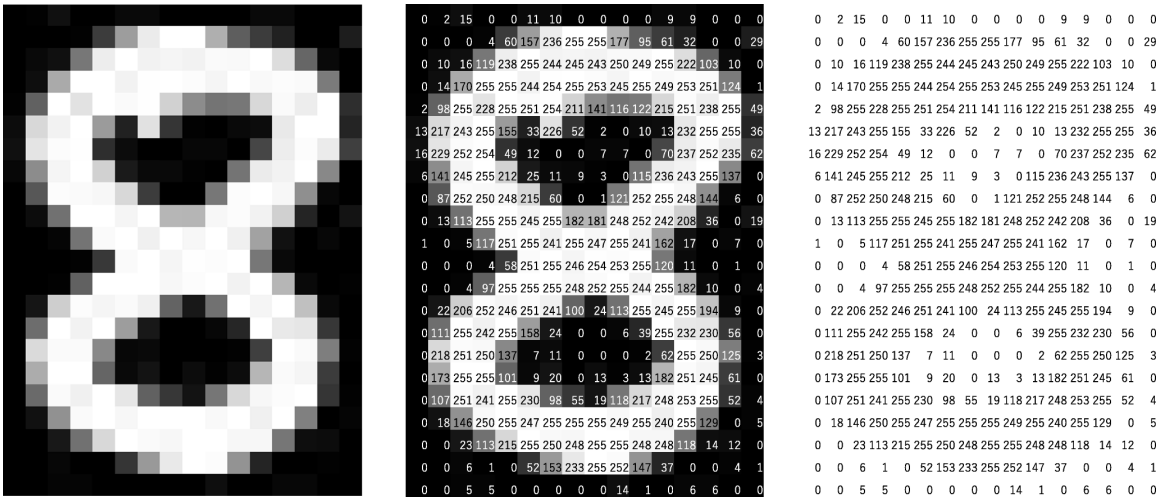

Figure 2.3: An image is seen by a computer as an array of numbers. There are numbers between 0 and 255 in the matrix on the right, and everyone corresponds to the brightness of a pixel in the image on the left. In the center image, both are overlaid [3].

fully connected layer maps the extracted features to the model output. Standard architectures have iterations of a stack of multiple pooling and convolution layers, and then a one or more fully-connected layers (see Figure 2.4). The process in which the input data is transformed across these layers into output is called forward propagation.

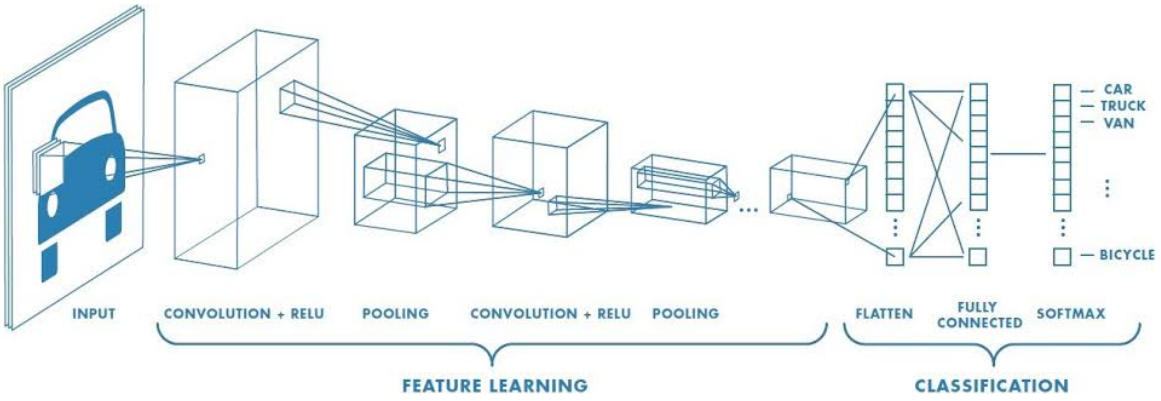

Figure 2.4: A typical convolutional neural network architecture with repetitions of convolution + ReLU + pooling layers for feature learning followed by fully connected layers for classification [4].

2.1.1 Convolutional layer

Convolution is one of the main building blocks of a CNN architecture. The feature extraction is done by a convolution layer, typically composed of a convolution operation and an activation function.

Convolution is a specific linear operation used for the extraction of features, where a kernel or filter feature extractor is applied throughout the data, which is a 2D number array (see Figure 2.3). Generally, convolution refers to the mathematical combination of two functions to produce a third function. In the case of CNN, convolution is done with the use of a kernel on the input data to create a feature map. A convolution is executed by sliding the kernel over the input and sums the result onto the feature map (see Figure 2.5). This procedure is repeated applying multiple kernels to form an arbitrary number of feature maps, which represent different characteristics of the input. Two key hyperparameters needed for the convolution operation are the step size and number of kernels. The size of a kernel is usually 3×3. The number of kernels is variable, and controls how deep the output feature maps are.

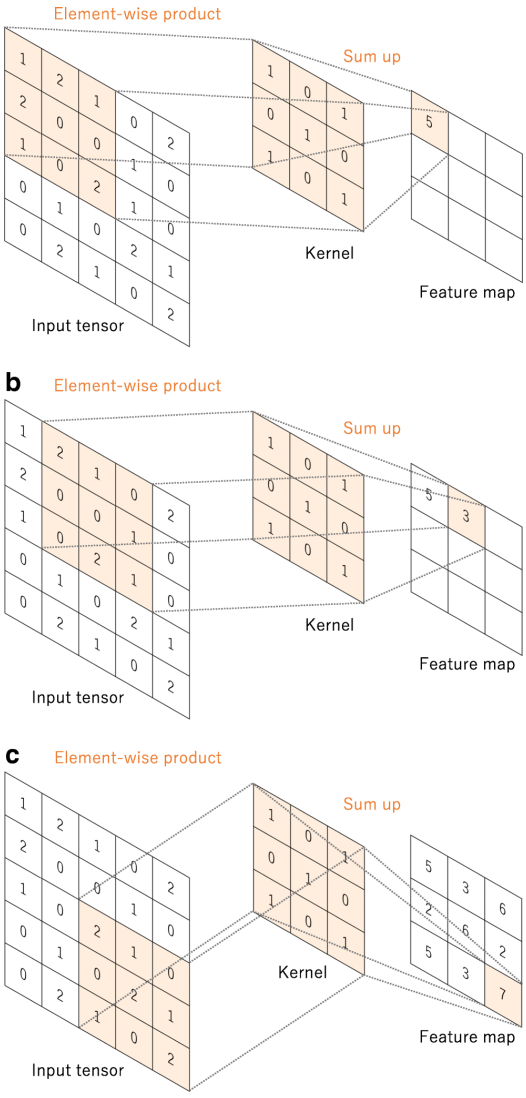

Figure 2.5: An instance of a convolution with a kernel size of 3×3, no padding, and a stride of 1 [3].

In the convolution operation mentioned above the edges of the input element do not overlap with the center of each kernel and therefore the size of the resulting feature map is reduced relative to the input tensor size (see Figure 2.5). Padding is a method used to combat this problem, where rows and columns of zeros are added to each edge of the input in order to match the center of the kernel to the edges and to preserve the same size through the convolution operation (see Figure 2.6).

The distance between the two consecutive positions of the kernel is known as stride, which also determines the result of the convolution operation. The typical selection of a stride is 1, but a stride greater than 1 is often chosen to downsample the feature maps. Pooling is another method for downsampling.

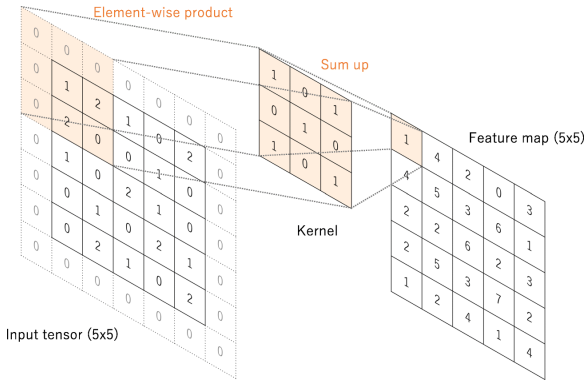

Figure 2.6: An example of a convolution operation with zero padding [3].

The outputs of a convolution operation are then passed through a nonlinear activation function. Smooth nonlinear functions have been the most common for many years. These include the hyperbolic tangent (tanh) and the sigmoid function. The rectified linear unit (ReLU) is by far the most popular nonlinear activation function currently used, which simply evaluates the function f(x) = max(0, x) (see Figure 2.7) [3].

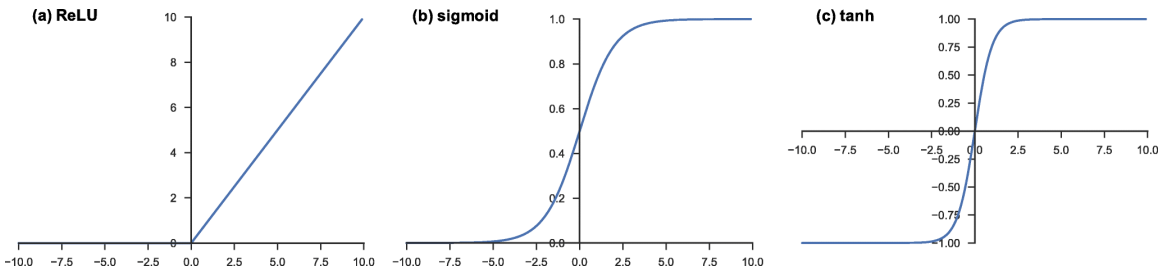

Figure 2.7: Popular activation functions currently used for CNNs [3].

2.1.2 Pooling layer

The pooling layer offers a standard downsampling functionality that decreases the spatial size of the input feature maps with the aim of adding invariance to minor shifts and distortions and to minimize the amount of successive learning parameters. Pooling acts a regulariser, while reducing the memory usage [3]. In any of the pooling layers there are no learning parameters; the filter size, phase and padding are hyperparameters in pooling operations [3].

The most common method of pooling is max pooling with a 2×2 filter and a stride of 2. The max pooling operation outputs the maximum value from an extracted set of points from the input feature maps [3] (see Figure 2.8). This operation reduces the spatial dimension of feature maps by a factor of 2. The depth of the feature maps remains unaffected, unlike width and height.

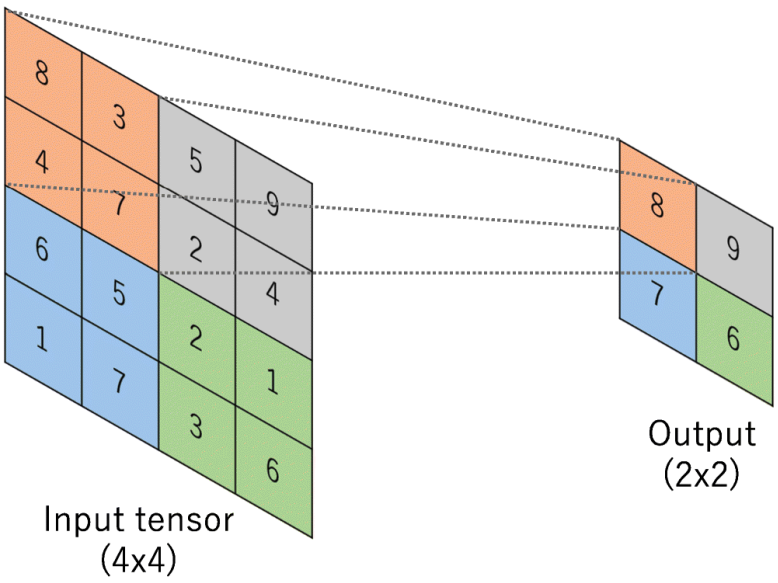

Figure 2.8: An instance of a max pooling operation with a 2×2 filter size, no padding, and a stride of 2 [3].

Global average pooling is another popular pooling operation. A feature map is downsampled into a 1×1 array in this operation by calculating the average of all the points on every feature map [3]. The depth of feature maps remains unaffected. Global average pooling is usually used right before the fully connected layers. This operation has the following advantages: it decreases the number of trainable parameters and allows the model to work with inputs of variable sizes [3].

2.1.3 Fully connected layer

A fully-connected layer is analogous to the way that neurons are arranged. Each node in a fully-connected layer is connected to every other node in the previous and in the next layer. Fully-connected layers usually have the most parameters within a CNN, and take a long time to train [3]. Therefore, to eliminate the number of nodes and connections the dropout technique is usually used, which is a regularization technique for neural network models proposed by Srivastava et al. [25]. A nonlinear function, such as ReLU, follows every fully connected layer. The feature maps from the last convolution layer are usually converted into a 1D array and connected to fully-connected layers [3].

The final fully-connected layer usually uses an activation function that is different from the ones used in all other layers. The best activation function depends on the task at hand. For example, a classification task with multiple classes usually needs a softmax function [3].

2.2 Training a CNN

The training process involves obtaining weights for fully-connected layers and kernels from convolution layers, to reduce deviations between ground truth labels and predicted output labels. During the training process kernels are the only parameters in convolution layers. Hyperparameters, which include padding, stride, kernel size, and number of kernels, need to be defined before the training begins. For pooling layers there are no parameters and the pooling method, filter size, and padding are hyperparameters. Fully-connected layers have weights as parameters, and the number of weights and activation function as hyperparameters. Other hyperparameters include weight initialization, batch size, learning rate, loss function and optimizer [3].

Backpropagation is a standard technique for training CNNs where the optimization function and loss function are critical components. The loss function calculates the model performance (accuracy and error) from the kernels and weights via forward propagation of the training data, and trainable parameters are updated and optimized through backpropagation.

2.2.1 Loss/cost function

The loss/cost function, alternatively known as the objective function, is a metric used to measure the model's performance. The loss function measures the difference between predicted labels from the model and given ground truth labels. The type of loss function used

depends on the given task. Cross entropy is the most popular loss function for classification tasks with multiple classes, and the mean squared error loss function is usually used for regression to continuous values.

2.2.2 Gradient Descent

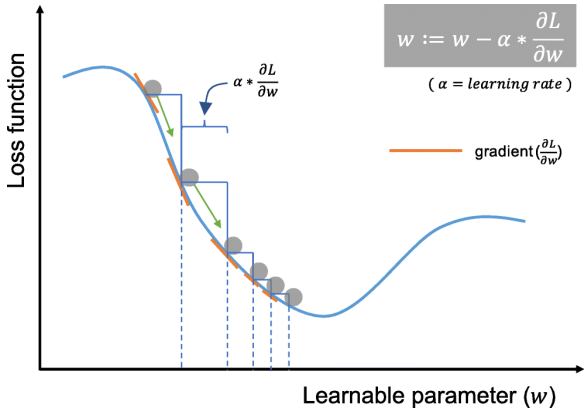

Figure 2.9: An illustration of the gradient descent optimization technique [3].

Gradient descent is widely utilised as an optimization technique to optimize the network's trainable parameters to reduce the loss of the network. The loss function gradient produces the direction where the function will have the highest rate of increase, and each training parameter is updated with a step size calculated on the basis of a hyperparameter called the learning rate (Figure 2.9) in the negative direction of the gradient. The gradient is a loss derivative for each trainable parameter, and a single parameter update is formulated as:

$$w := w - \alpha \times \frac{\delta L}{\delta w},$$

where w stands for each parameter that can be learned, α is the learning rate, and L is the loss function. In practice the learning rate is one of the most important hyperparameters. In reality, due to memory constraints the parameter gradients of the cost function are determined using a mini-batch which consists of a subset of the training data. This approach is referred to as mini-batch gradient descent, or sometimes stochastic gradient descent (SGD). The mini-batch size is not a trainable parameter. Several enhancements have been suggested and commonly used in the gradient descent algorithm. These include Adam [26], RMSprop [27], and momentum SGD [28].

2.3 Hardware and software

The abundant availability of GPUs and GPU computing libraries (such as OpenCL and CUDA) has been one of the key reasons for the recent growth of deep learning applications. GPUs are highly parallel computing engines that are an order of magnitude faster than central processing units (CPUs). With state-of-the-art hardware, GPUs are usually 30 times faster than CPUs for deep learning applications.

- **CUDA** is a computing platform and programming parallel platform developed by Nvidia on its own GPUs for general computing. By harnessing the power of GPUs for the parallelizable part of a computation, CUDA helps developers to speed up compute-intensive applications.

- **OpenCL** is a framework for writing programs that operate across heterogeneous systems consisting of central processing units, graphics processing units, digital signal processors (DSPs), field-programmable gate arrays (FPGAs) and other hardware accelerators or processors.

Next to hardware, the large presence of open source software packages is the other driving force behind the success of deep learning approaches. These libraries provide effective GPU implementations of critical neural network operations such as convolutions, enabling users to incorporate projects at a high-level instead of thinking about low-level effective implementation [29]. Public accessibility to programming frameworks that apply the key strategies in this field in high-level programming languages has partially allowed the growth of deep learning. Software developers are currently managing all of these systems on an ongoing basis, and most of the latest findings are easily integrated into them. Although the availability of a suitable graphics processing unit is needed to take full advantage of these modern architectures, most of them are often equipped with CPU support to train and test small models. Without the need to actively execute the operations performed by the layers and the algorithms that train them, the frameworks allow their users to directly evaluate various network architectures and their hyperparameter settings. The layers and corresponding algorithms are already implemented in the frameworks' libraries. We list the most common deep learning frameworks below. We order them, beginning with the most popular, based on their present day importance in the computer vision and pattern recognition community for the problems of image processing:

- **Tensorflow** [30]: This software library for machine learning is free and open-source. It can be used across a variety of activities but has a special emphasis on deep neural network training and inference. It provides an interface between Python and C++.

- **Torch** [31]: This is a library for open-source machine learning, a platform for scientific computing, and a programming language based on the Lua script language. It offers a wide variety of deep learning algorithms and uses LuaJIT, the scripting language, and an underlying C implementation.

- **PyTorch** [32]: This is a Torch library-based open source machine learning library, used for applications such as computer vision and natural language processing, mainly developed by the AI Research lab of Facebook. It is software free of charge and open-source.

- **Theano** [33]: This is a Python library and compiler optimization for manipulating and evaluating mathematical expressions. Computations are represented in Theano using a NumPy-esque syntax and compiled on either CPU or GPU architectures to run efficiently.

- **Caffe** [34]: This was originally developed at the University of California, Berkeley, as a deep learning system. Under the BSD license, it is open source. It is written in C++, with an interface in Python.

- **Caffe2** builds on Caffe and provides C++ and Python interfaces.

- **Keras** [35]: This is an open-source library for artificial neural networks that provides a Python interface. Keras serves as a TensorFlow library interface. Multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, R, Theano, and PlaidML, were provided by Keras until version 2.3.

- **MatConvNet** [36] is a MATLAB toolbox implementing convolutional neural networks for computer vision applications.

This is not an extensive compilation of deep learning frameworks. It does, however, cover certain structures that are commonly used in image processing at present. It goes beyond this dissertation's reach to address in depth all these frameworks.

Chapter 3

Deep learning in medical imaging

A significant part of the radiological workflow is the classification and detection of anatomical structures in medical images. These tasks are typically performed by radiologists through identifying certain anatomical signatures, specifically image features that can differentiate between one anatomical structure and others [12]. New research has shown that, for two reasons, deep learning-based approaches have become prevalent: deep learning systems are now powerful enough to solve real-world problems, and more medical image datasets are available to promote the study of massive medical image data [12]. One of the earliest domains where deep learning had a significant contribution to medical image processing was image classification. An exam classification usually has one or more images as the input and one diagnostic element as the output (e.g. illness present or not).

Due to the expense and required workload of radiology professionals, an availability of correctly labelled medical imaging data is ideal but seldom present. A few techniques are available for training a model effectively on a smaller dataset, namely, data augmentation and transfer learning [3]. Transfer learning has been central to several medical imaging systems [18], where the current practice is to use an existing architecture created for natural image datasets like ImageNet [2], along with suitable pre-trained weights (e.g. ResNet [37], Inception [38]), and then fine-tune the medical imaging data model. However medical images and natural images are significantly different and transferring weight or kernels from ImageNet might not be effective.

This chapter provides a review of popular deep learning based classification focusing on convolutional neural network approaches, followed by a brief discussion about the availability of datasets for medical imaging tasks. We also discuss how medical images differ from natural images (e.g. features, data sizes, and task specifications) and therefore introduce the question of the effectiveness of the current approach of transferring representations

learned from natural images for medical imaging tasks. We also provide a background on autoencoders.

3.1 Transfer learning overview

Transfer learning is a standard and powerful technique for training a deep learning architecture with a small dataset. In transfer learning a deep learning network is pre-trained on an incredibly huge dataset, like ImageNet, which contains more than 14 million images in over 20,000 categories, then retrained for the target task. The basic principle of transfer learning is that non-specific features learned from a large enough dataset can be spread across seemingly different domains [3]. This transferability of trained generic features is a distinctive benefit of deep learning that, with limited datasets, makes itself useful in different domain tasks. At present, along with their learned kernels and weights, several models pre-trained on the ImageNet challenge dataset are available to the public, such as AlexNet [2], VGG [7], ResNet [37], Inception [6], and DenseNet [8].

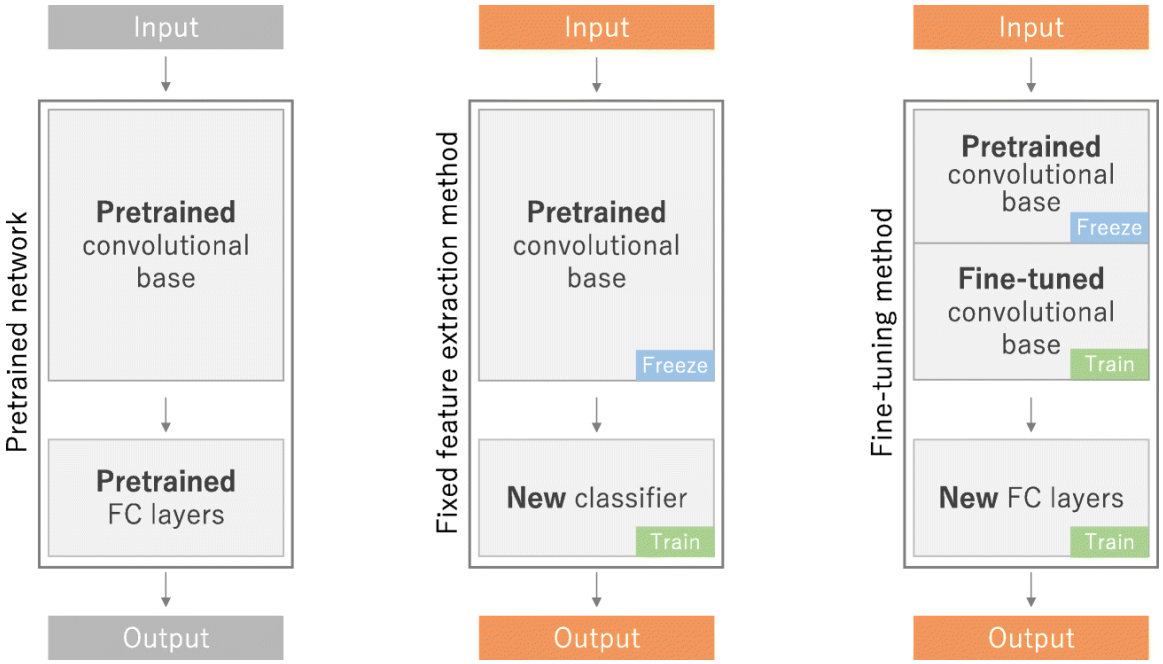

Figure 3.1: Transfer learning strategies. The fixed feature extraction method (middle) freezes the pretrained convolutional base from the pretrained network (left) and trains a new classifier. The fine-tuning method (right) freezes a fraction of the convolutional base and trains the rest along with the classifier [3].

Fixed feature extraction is a method of replacing fully-connected layers (classifier) from a network pre-trained on ImageNet and retraining the new fully-connected layers and freezing the rest of the network (see Figure 4.5). In this example any classic classifier (e.g random forests, support vector machines) and also the normal fully-connected layers in CNNs, can also be applied after the fixed feature extractor, leading to training limited to the applied classifier on a given dataset of interest. Due to the dissimilarity between ImageNet and medical images, this method is not popular in deep learning applications on medical images [3].

A more commonly used fine-tuning method for radiology study is not just to exchange fully connected layers from the pre-trained model with a new set of fully connected layers in order to retrain for the particular target dataset, but also to retrain all or some of the kernels of the pre-trained convolution base via backpropagation (see Figure. 4.5). It is possible to retrain all the layers in the convolutional base or, alternatively, to fix certain earlier layers while retraining the rest of the CNN layers. This method is informed by the finding that features in early layers, including features such as corners and edges that apply to a number of datasets and tasks, appear more general, while features from later layers increasingly become unique to a specific dataset or task [18].

3.1.1 Transfer learning metrics

The are many different metrics to measure transfer learning performance. In Figure 3.2, common parameters for measuring transfer learning performance are shown. The difference between the initial values, with and without transfer learning, is called jump-start. The final performance of the model is called asymptotic performance and the time required to achieve a pre-defined level is called time to Threshold or Speedup. These metrics are explained in detail in [5].

In this work we only measure asymptotic performance.

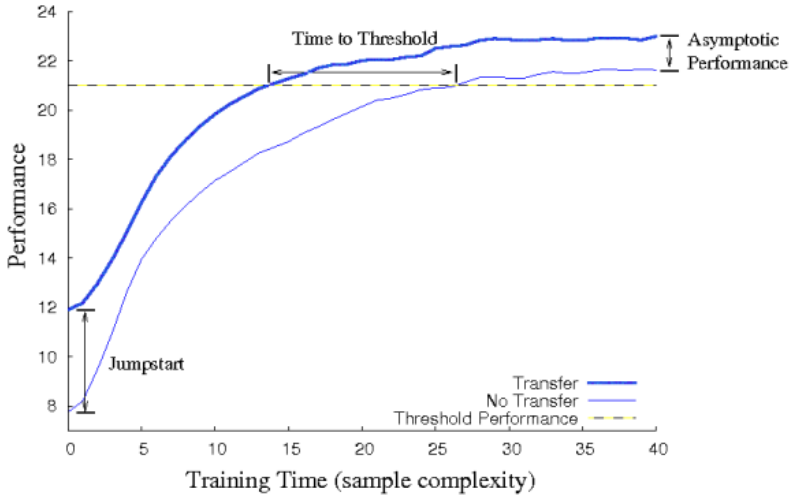

Figure 3.2: Common metrics for measuring transfer learning performance. This graph show the jump-start, asymptotic performance, time to threshold or speedup [5].

3.2 Architectures for classification

In this section we review popular ImageNet architectures used in medical imaging, focusing on X-ray images. Our review focus on Inception, VGG and DenseNet architectures because these make up majority of the application of ImageNet models on medical images [39].

3.2.1 InceptionV3

In 2015 Szegedy et al. [6] introduced Inception-v3 which had improvements on previous versions of the inception family architectures (Inception-v1 [40] and Inception-v2 [6]). The improvements included factorized 7×7 convolutions, label smoothing, and the use of an auxiliary classifier to spread label information lower down the network. The model is made up of standard convolutional neural network building blocks, which include convolution, average pooling, max pooling, dropouts, concatenations, and fully-connected layers (see Figure. 3.3). Batch normalisation is used extensively throughout the model and applied to activation inputs. Loss is computed via softmax. The Inception module's main concept is to use multiple size filters with distinct receptive fields to capture local variability and thus enhance characterization. Inception models (in particular the Inception-v3) were the most popular among the studies that worked with skeletal X-rays (i.e. wrist, hip and knee), making up about 57% of studies [39]. For X-rays in general Inception-v3 accounts for about 30% of studies [39].

Kim et al. [41] retrain the last layer of the Inception-v3 model on lateral wrist X-rays

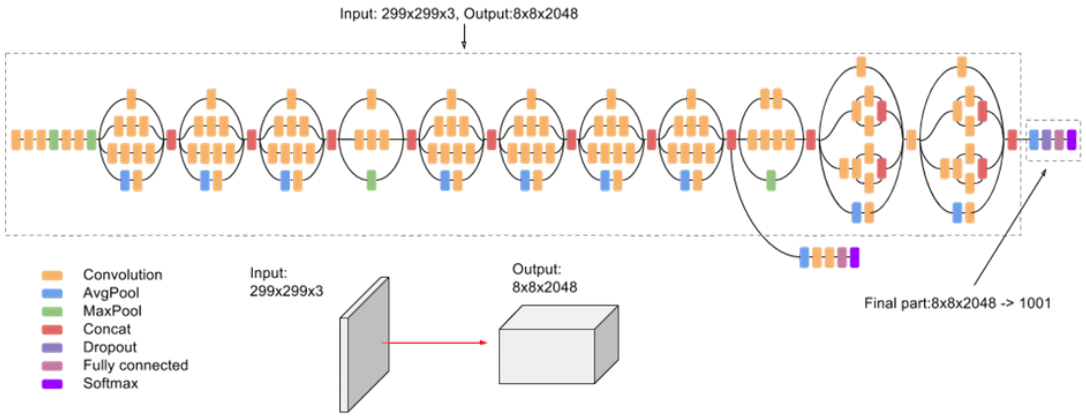

Figure 3.3: Inception V3 architecture [6].

for the classification of X-rays images as either having a 'fracture' or 'no fracture'. They use an 8-fold data augmentation technique on a dataset of 1389 images, resulting in a training dataset of 11112 X-rays. They report an area under the receiver operating characteristic curve (AUC) of 0.954, a sensitivity of 0.9, and a specificity of 0.88.

Yu et al. [42] use an Inception-v3 model to classify fractures in the hip. The output layer of the Inception-v3 was replaced by a fully connected layer, which was connected to another fully connected layer (with a softmax activation) with two nodes through a dropout layer. Each layer uses a ReLU activation. They report an AUC of 0.994, with a sensitivity of 0.971, and a specificity of 0.997 for fracture detection.

Lee et al [43] use 3000 periapical X-ray images to pre-trained Inception-v3 for detection and diagnosis of dental caries, with the training set containing 2400 X-ray images and the test set containing 600 X-ray images. All images were resized to 299×299. They achieve 89% in accuracy for the molar model, 88% for the premolar model and for both models the achieve 82%. The AUC achieved by the Inception-v3 network for premolar, molar, and both premolar and molar models was 0.917, 0.890, 0.845 respectively.

3.2.2 VGG

VGG-16 and VGG-19, two popular ImageNet architectures, were first implemented by the Visual Geometry Group (VGG) at ILSVRC2014 [7]. These models outperform AlexNet [2] by replacing large kernel-sized filters with several small kernel-sized filters, resulting in VGG-16 (see Figure 3.4) and VGG-19 having 13 and 16 convolution layers, respectively.

VGGNet accounts for about 20% of studies done for X-rays [39].

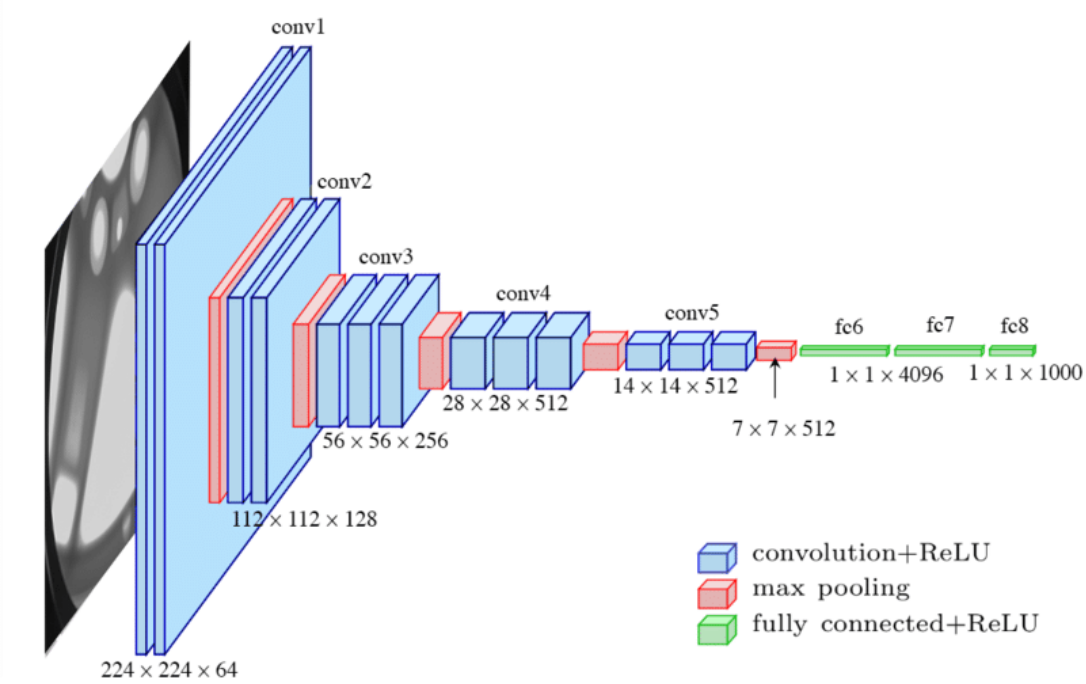

Figure 3.4: VGG-16 architecture [7].

In their study, Ahsan et al. [44] apply the VGG-16 architecture on a chest X-ray dataset to identify tuberculosis (TB) from X-ray images. The dataset used contained X-ray images from Montgomery (1324 X-ray images) and Shenzen (276 X-ray images) datasets. Images were resized to 224×224. Their model was able to achieve an accuracy of 80% without using data augmentation and 81.25% with the use of data augmentation.

Prajapati et al. [45] use a pretrained VGG-16 as a feature detector for dental disease classification. The classification task is distributed into three classes which are dental caries, periapical infection and periodontitis. On a limited dataset of 251 dental RVG X-ray images, transfer learning with the pretrained VGG-16 model is used. The images are resized to 224×224 and fed into the network as an input. The weights for VGG-16's first two convolution layers were frozen, and the rest of the layers were used for training. Out of a total of number of X-ray images, the training split had 180 X-rays, validation split had 45 X-rays, and the test split had 26 X-rays. A total accuracy of 88.46% was obtained.

Poedjiastoeti et al. [46] use a transfer learning approach to deal with the problem of

limited image data for the diagnosis of jaw tumours. They used a dataset with 500 X-ray images, with a training set of 400 images and a test set with 100 images. They use a pre-trained VGG-16 network, evaluate the performance of VGG-16 compared to diagnosis made by maxillofacial and oral specialists. The VGG-16 model achieved an accuracy 83%, a sensitivity of 0.818, a specificity of 0.833, and a diagnosis time of 38 seconds. The specialist achieved an accuracy 82.9%, a sensitivity of 0.811, a specificity of 0.829, and a diagnosis time (time to reach a diagnosis) of 23.1 minutes.

3.2.3 DenseNet

Each convolution layer in DenseNet receives as input the output (i.e. feature maps) of all previous layers and transfers its own output to all subsequent layers [8]. As a result, each layer gains access to the collective information of all previous layers. Since the number of feature maps is reduced, the resulting CNN model becomes smaller and more compact. DenseNet comes with a variety of versions, including DenseNet-169, DenseNet-121 (see Figure 3.5), and DenseNet-201 [8]. DenseNet accounts for about 20% of studies done for X-rays [39].

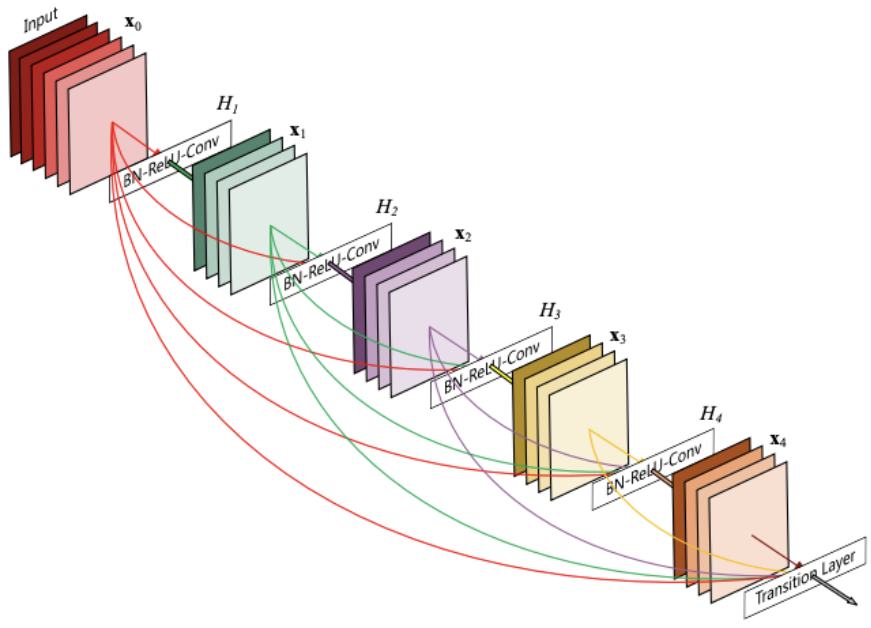

Figure 3.5: DenseNet-121 architecture [8].

Nguyen et al. [19] investigate the usefulness of transfer learning from natural images on medical imaging for tuberculosis detection. They pretrain a DenseNet-121 with the Na-

tional Institutes of Health (NIH) dataset, which has 112120 chest X-ray images with 14 different classes of common chest illnesses. For the tuberculosis classification they use the Montgomery and Shenzhen datasets. They show pre-training the DenseNet-121 on the NIH dataset produces better performance than pretraining from ImageNet. They also show that low-level features from ImageNet weights are inadequate for imaging tasks for modalities such as X-rays, and that using suitable data for pre-training is necessary and makes the whole process more effective.

3.3 Availability of datasets

Significant volumes of training datasets are the first and major requirement for the use of deep learning as the accuracy and evaluation of deep learning-based classifiers rely heavily on the quality and quantity of data. The greatest barrier to the progress of deep learning in medical imaging is the limited supply of medical imaging data.

The creation of a vast training dataset is itself a time-consuming and laborious process that requires a significant amount of input from medical professionals. Therefore, the production of quality data on a large scale, especially for rare diseases, needs more trained experts. In addition, for deep learning algorithms to effectively learn the representations, a balanced dataset is required. Many of the available datasets are unbalanced in healthcare, leading to class imbalance.

3.4 Medical images versus natural images

Transfer learning is normally carried out, as mentioned in Section 3.1, by taking a regular ImageNet model with pre-trained kernels and weights, and then retraining for the target task. Natural image classification with ImageNet models and medical image analysis, however, have major variations [18].

Medical imaging activities often begin with a wider view of an area of interest in the body and use local texture differences to recognize pathologies. For instance, tiny red spots are an example of diabetic retinopathy and microaneurysms in retinal fundus images, and local white cloudy areas are indicators of consolidation and pneumonia in chest X-rays [18]. This is unlike natural image datasets such as ImageNet, in which a clear global image object is always present. Thus, there is an unanswered question about how much reuse of ImageNet functionality is useful for medical images. In comparison, most medical datasets have

bigger images, but with far fewer images than ImageNet, which has over 14 million images. Medical datasets, by comparison, vary from a few thousand to a few hundred thousand images.

Diagnostic activities also have substantially fewer classes than the traditional ImageNet classification of 1000 classes (5 classes for diagnosis of diabetic retinopathy, and 5 to 14 for identifying chest pathologies from X-rays). For precisely this reason, typical ImageNet models have a large number of parameters clustered in the higher layers. The configuration of these architectures is likely to be sub-optimal for the medical context.

Raghu et al. [18] examine transfer learning properties for medical imaging. They show that transfer learning from natural images (i.e. weights and kernels of the ImageNet model) provides minimal performance gain, and plain, lightweight models can perform similarly to ImageNet architectures. Instead of advanced feature reuse, they found that some of the variations from transfer learning were due to the over-parametrization of standard models.

3.5 Background to Autoencoders

An autoencoder is a neural network whose aim is to reconstruct its input with one or more hidden layers. The autoencoder is required to learn how to best represent the input data compactly by adding a hidden layer of very few nodes, commonly referred to as the representation layer. Features are given by the representation layer until the network is trained. Figure 3.6 provides a graphical representation of a regular autoencoder.

Two parts of the network can be considered: the encoder h, that translates the input vector x to the latent space representation $h(x)$, and the decoder g, that seeks to translate the latent space representation back to input x. The autoencoder is programmed to optimize a cost function that penalizes the error of the reconstruction which is an estimate of the uncertainty between input x and the predicted output $g(h(x))$ of the subsequent reconstruction. The denoising autoencoder, which is trained on input data with noise, is a common type of autoencoder, penalizing the error of reconstruction against the noiseless originals of the reconstructions. The contractive autoencoder, another variant, explicitly penalizes the autoencoder's vulnerability to disturbances in the input.

Autoencoders aim to obtain a lower dimensional representation from the input without losing large quantities of information. Therefore, autoencoders inevitably operate under the premise that there are lower dimensional manifolds on which the data lies in the input

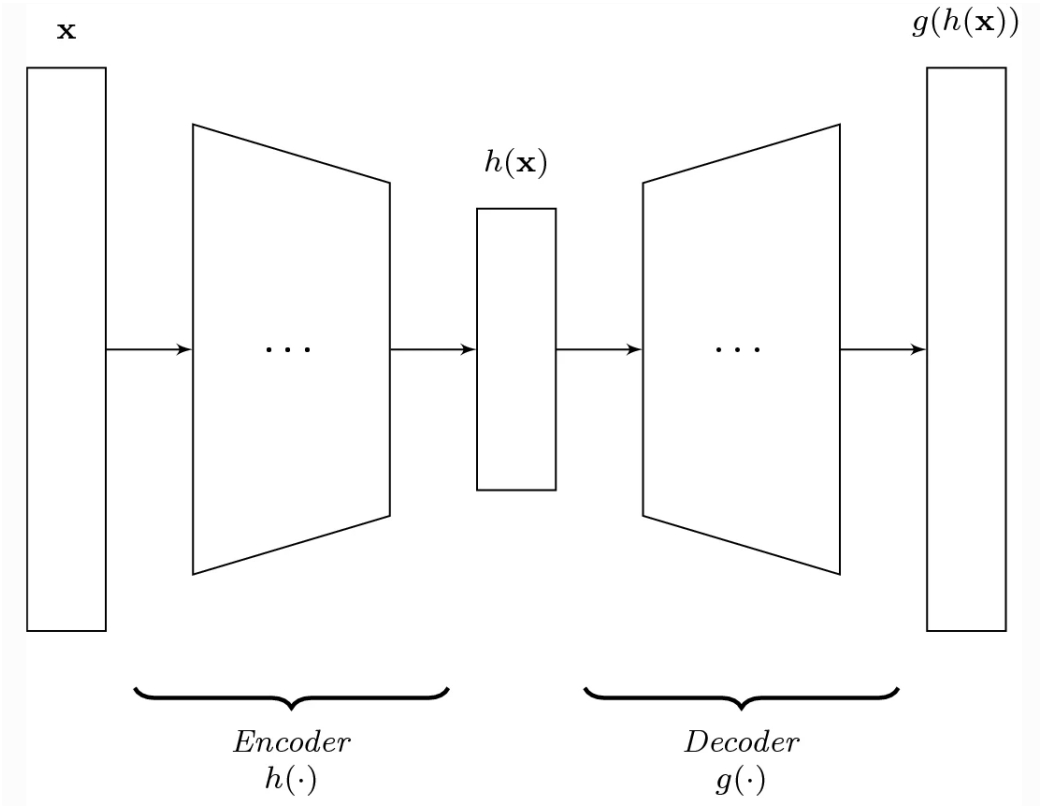

Figure 3.6: A basic illustration of an autoencoder.

space. Furthermore, they presume that two samples on the same lower dimensional substructure have the same label when added as a preprocessing step to classification. These findings suggest that the theories fundamental to autoencoders are closely related to the semi-supervised multiple manifold assumptions [47].

In semi-supervised learning, the manifold assumption states that the input consists of several lower dimensional manifolds on which all data points are located, and data points have the same label on the same manifold. As a result, if we can determine that the manifolds exist and which data points lie on which manifold, it is possible to assume class assignments of unlabelled data points from the classified data points on the same manifold.

In the section we give a brief background on the autoencoder with a focus on convolutional autoencoders. We also provide a review of autoencoder applications in medical imaging with a focus on unsupervised learning.

3.5.1 Convolutional autoencoders

It is normal to use convolutional layers for an image autoencoder because convolutional neural networks perform well at several computer vision tasks. Pooling layers are typically used with convolutional layers in convolutional autoencoders to minimize the size of the hidden representation layer. In the encoder, the hidden layer is often followed by a fully-connected layer, which is reshaped to the proper size before the decoding stage. The decoder needs to resize the hidden layers so the convolutional autoencoder's output is the same size as the input. This can be done in two ways:

- Before each convolutional layer, upsampling the hidden layer, e.g. with bilinear interpolation.
- To perform a trainable method of upsampling, advanced transposed convolution layers are used.

The second way is more principled and normally produces better outcomes, but it also increases the number of network parameters and may not be optimal for all types of problems, particularly if there is insufficient training data.

3.5.2 Applications of autoencoders

Stacked autoencoders are the most common and widely used architecture in unsupervised learning [47]. Layer-by-layer pre-training is required for a stacked autoencoder. This is because a stacked autoencoder is created with fully-connected layers, and going deeper into the layers throughout pre-training can take too much time. Li et al. [48] present the first trial in which convolutional neural networks are trained end-to-end without any pre-training. Guo et al. [49] proposed the convolutional autoencoder (CAE), which is useful for learning image features while keeping the data's local structure and avoiding feature space distortion.

A sparse convolutional autoencoder is proposed by Hue et al. [50] for simultaneous nucleus recognition and feature extraction in histopathology tissue images. Their convolutional autoencoder recognizes nuclei in image patches in tissue pictures and converts them into sparse feature maps that encode nuclei's position and appearance.

Hosseini et al. [51] proposed a 3D Adaptable CNN classifier that outperforms multiple existing state-of-the-art systems for predicting Alzheimer's disease on structural brain MRI scans. To improve the generality of features in capturing Alzheimer's Disease biomarkers,

pretraining and layer freezing were applied. The 3D convolutional autoencoder network is pre-trained on the CADDementia dataset. The learned features are then retrieved and applied in the bottom layers of a 3D Adaptable CNN network to detect Alzheimer's Disease biomarkers. Then, on top of the bottom layers, three fully connected layers are stacked to perform Alzheimer's Disease classification on 210 ADNI dataset individuals.

3.6 Key points

In this chapter we review popular architectures for solving the classification problem for medical images. ImageNet architectures pretrained from natural images are standard for medical imaging due to the lack of large-scale labelled medical datasets. The most popular ImageNet architectures are Inception-v3 [6], VGG-16 [7], and DenseNet-121 [8].

However, due to the differences between natural and medical images we expect features from ImageNet to not be the most effective in medical imaging tasks. Nguyen et al. [19] investigate the usefulness of transfer learning from natural images on medical imaging for tuberculosis detection and show that pre-training a DenseNet-121 model with 112000 X-ray images performs better than when pretrained with ImageNet. Raghu et al. [18] also show that transfer learning from natural images (i.e. weights and kernels of the ImageNet model) provides minimal performance gain, and plain, lightweight models can perform similarly to ImageNet architectures. Hence, we propose feature learning from unlabelled X-ray images (using an autoencoder) instead of natural images and pre-training lightweight architectures instead of large ImageNet architectures.

Chapter 4

Proposed method and network architecture

Transfer learning utilises features learned for one task to improve other tasks. The learning process can be faster, more accurate and more importantly may require less training data. This makes transfer learning attractive for deep learning for medical imaging, due to the limited availability of labelled medical imaging data. The standard practice in medical imaging is to use large-scale natural imaging datasets (e.g. ImageNet). However, medical images and natural images are significantly different. Developing a large-scale medical imaging dataset for transfer learning would be too expensive, hence the use of a large-scale unlabelled dataset is very attractive. We propose the use of an autoencoder to learn features from an unlabelled medical imaging dataset through a reconstruction, and then transfer those features to the target task (classification) to improve performance.

In this chapter we modify the CBR-LargeT architecture [18] into an autoencoder for the reconstruction task and use a slightly modified CBR-LargeT for the target task. We also describe the datasets used in experiments (namely ImageCLEF, GWCMPD and RSNA) and the preprocessing techniques used.

The chapter is structured as follows. The proposed method is presented in Section 4.1, the model architectures used are detailed in Section 4.2, and datasets and preprocessing techniques are described in Sections 4.3 and 4.4 respectively.

4.1 Proposed method

The concept underlying transfer learning for image classification is that once a model is trained on a sufficiently broad and diverse dataset, that neural network would essentially act as a comprehensive representation of the visual domain [18]. By pre-training a large enough CNN on a large-scale dataset, you could use these trained feature maps instead of starting from scratch. Thanks to the large-scale image datasets, e.g., the ImageNet [17] and Places [52] datasets, the source domain is often large enough to share similar representations with the target domain in both low-level and high-level features. Thus, models pretrained on these datasets often have good initialization, which enables them to surpass models trained from scratch in various open problems.

The limited availability of labelled medical imaging data introduces a huge obstacle to training CNNs due to the expense and workload required for radiologists to annotate medical images. Therefore, transfer learning from natural image datasets (i.e. using standard large ImageNet models and corresponding pretrained kernels and weights) has become a standard.

As discussed in Section 3.4, there are significant differences between medical and natural images (e.g. features, data sizes, and task specifications), and there is little knowledge of the consequences of transferring natural image learning to medical image tasks [18]. In addition, transfer learning from natural images has been shown to have very little performance gains [18] and transfer learning from a large enough medical imaging dataset (i.e. 112000 x-ray images) performs better that when pretrained with ImageNet [19].

The main drawback to supervised pretraining is that, like ImageNet, a large amount of costly manually labelled training data is necessary. Therefore, the prospect of using large quantities of unlabelled data for feature learning is very desirable, due to the limited availability of labelled medical imaging data and the significant differences between natural images and medical imaging. We propose using an autoencoder to leverage unlabelled data by transferring weights/kernels learned by the autoencoder (specifically the encoder part) during an unsupervised reconstruction task (from a large collection of unlabelled medical x-ray images) to a classification task. This method can have better performance than state-of-the-art ImageNet transfer learning.

The proposed method has two stages, the unsupervised learning stage (image reconstruction) and the supervised learning stage (classification target task). Firstly, we train the stacked autoencoder for a reconstruction task (using a source dataset of unlabelled images). After that we transfer the weights and kernels from the encoder to the classification model (see Figure 4.1), and then we train the classification model with the target dataset.

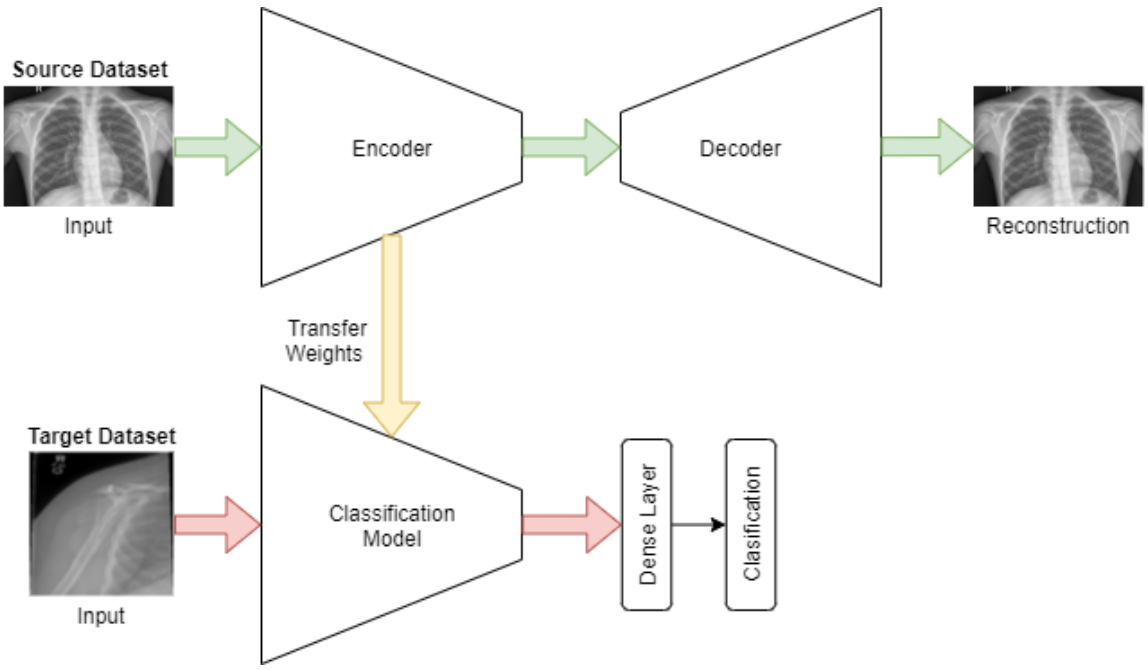

Figure 4.1: An illustration of our proposed method. The green path shows the reconstruction stage, where the stacked autoencoder is trained to reconstruct its input. The yellow path shows the transferring of kernel weights to the classification model. The red path shows how the classification model with pretrained kernels is retained for the target task.

4.2 Description of models

For the target task, we design a CNN model inspired by a group of non-standard tiny and basic architectures, namely the CBR models from Raghu et al. [18]. The CBR models are a group of CNN architectures consisting of several convolutional, batch normalisation and ReLU layers along with maxpool operations. For all CBR models, all the convolutions have the same filter size [18]. The chosen CBR model (CBR-LargeT) has 5 convolutional blocks (see the convolutional block description in Section 4.2.1). In the last convolutional, block we replace the maxpool with a global average pool layer followed by a fully connected classification layer (see Figure 4.2 for the final classification model). A detailed description of the CBR-LargeT model can be seen in Table 4.1.

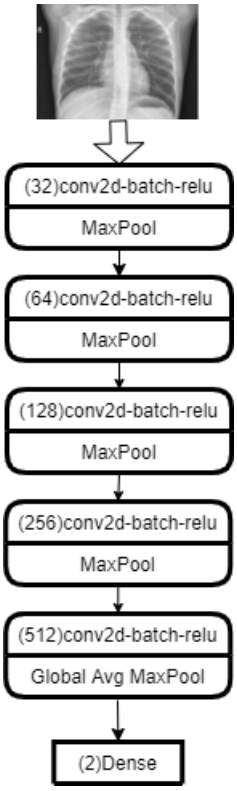

Figure 4.2: CBR-LargeT.

For the reconstruction task we extend the CBR-LargeT model into a stacked autoencoder. We use the first part of the CBR-LargeT model as an encoder part of the autoencoder. We remove the global average pool layer and fully connected layer and replace them with four upsample blocks (see the upsample block description in Section 4.2.2). The upsample blocks are followed by upsample, convolution (with the number of filters at 3), batch normalisation and ReLU layers (see Figure 4.3 for the final autoencoder model).

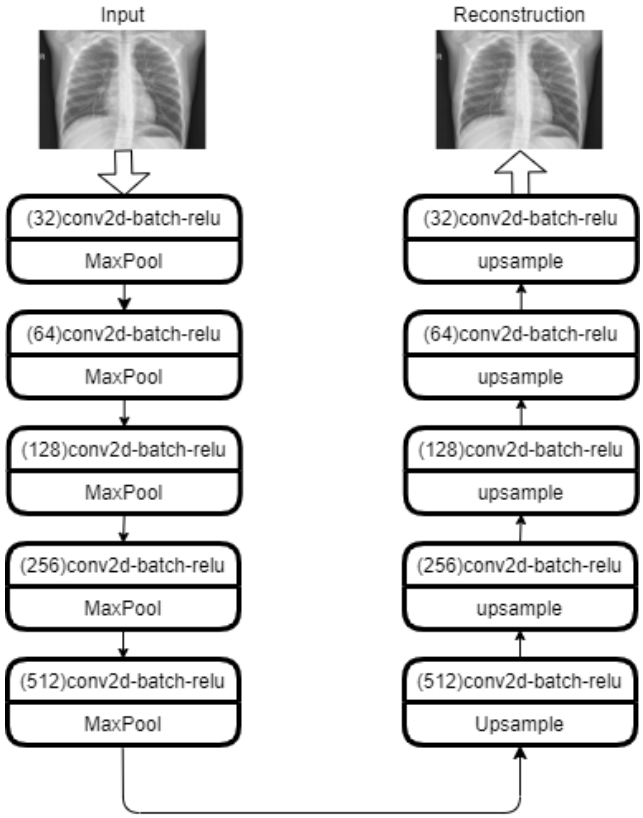

Figure 4.3: CBR-LargeT-AutoEncoder.

4.2.1 Convolutional block

The convolutional blocks consist of a convolutional layer, a batch normalisation layer, a rectified non-linear unit (ReLU) followed by a maxpooling operation. A description of the setup and hyper parameters of the block are as follows:

- **Convolution layers:** The number of filters starts at 32 and we double the number of filters after every maxpool operation. For the kernel size we use 3×3 for all convolutional layers and a stride of 2×2. The input feature maps are zero-padded on all sides so that spatial size is preserved through the convolution.
- **No setup is required for Batch normalisation and ReLU.**
- **Maxpool:** Each maxpool operation has a spatial window of 3×3 and a stride of 2×2.

Table 4.1: The CBR-LargeT has 3×3 convolution filters and has over 1.5 million trainable parameters. For the padding column 'valid' means no padding is done and 'same' indicates padding evenly on all sides of the input for the output to have the same dimensions as the input.

Layer	kernel Size / pooling size	# of filters	stride	padding
Conv2D	3×3	32	1	same
MaxPooling2D	3×3	-	2	valid
Conv2D	3×3	64	1	same
MaxPooling2D	3×3	-	2	valid
Conv2D	3×3	128	1	same
MaxPooling2D	3×3	-	2	valid
Conv2D	3×3	256	1	same
MaxPooling2D	3×3	-	2	valid
Conv2D	3×3	512	1	same
	number of units	activation		
GlobalAveragePooling2D	-	-	-	-
Dense	2	softmax		

4.2.2 Upsample block

The upsample block consists of a convolutional layer, a batch normalisation layer, a rectified non-linear unit (ReLU) followed by an upsample layer. A description of the setup and hyper parameters of the block are as follows:

- **Convolution layers:** number of filters starts at 512 and we halve the number of filters after every upsample step. For the kernel size we use 3×3 for all convolutional layers, and a stride of 2×2. The input feature maps were zero-padded on all sides so that spatial size is preserved through the convolution.
- **No setup is required for Batch normalisation and ReLU.**
- **Upsample:** Each upsample layer uses bilinear interpolation and has an upsampling factor of 2.

4.3 Datasets

For our source dataset we use the ImageCLEFmed 2009 dataset, and for the target dataset we use the Guangzhou Women and Children's Medical Center Pediatric Dataset. Brief descriptions of the datasets are provided below.

Source Dataset: The ImageCLEFmed 2009 dataset is a collection of anonymous radiographs selected from the Department of Diagnosis Radiology, at the Aachen University of Technology (RWTH), Germany. The dataset consists of images taken from patients of different ages and genders, resulting in a highly diverse collection. All images in the dataset are downscaled to fit into a 512×512 bounding box.

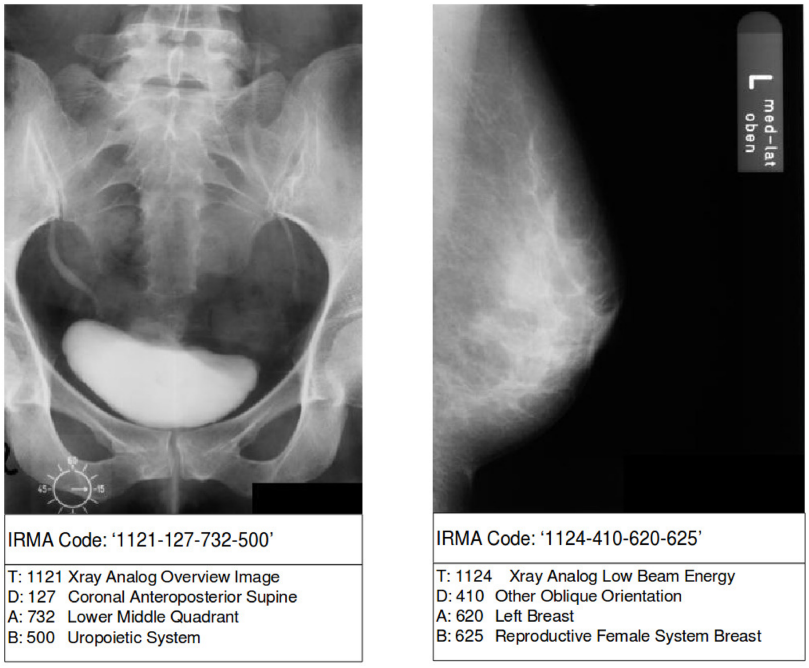

Figure 4.4: Sample images from ImageCLEF 2009 dataset with their respective IRMA codes [9].

Target Dataset: The Guangzhou Women and Children's Medical Center Pediatric Database is a publicly available database from Guangzhou Women and Children's Medical Center (GWCM) in China containing 5856 pediatric chest radiographs. Chest radiographs in this database were either labelled as normal or pneumonia (caused by virus or bacteria).

Figure 4.5: Examples from Guangzhou Women and Children's Medical Center Pediatric dataset showing patients with viral and bacterial pneumonia [10].

4.4 Data preprocessing

In this section we describe the methods used to preprocess the datasets used in all experiments.

4.4.1 ImageCLEFmed 2009

To transform the grey-scale images in the ImageCLEFmed 2009 dataset to 3-channel images for ImageNet architectures, padding techniques and layering are used to prepare and preprocess the x-ray images. Figure 4.6 shows an x-ray image from the dataset that will be used to demonstrate each technique's impact.

Figure 4.6: X-ray image from ImageCLEFmed 2009 before preprocessing [11].

4.4.1.1 Padding

Since the images in the ImageCLEFmed 2009 dataset are of different sizes, padding is used to resize them to size 320×320. As seen in Figure 4.7, the images are first padded with a constant value (using OpenCV's BORDER CONSTANT) that fills surrounding pixels with zeros.

Figure 4.7: Constant padding [11].

4.4.1.2 Layering

For the ImageNet pretrained models, images in the ImageCLEF dataset must be translated to 3-channel RBG images. To generate the RBG images, we adopted a similar procedure to [9]. The first layer was obtained from the CLAHE (a contrast enhancement method, modified from the Adaptive Histogram Equalization (AHE)) output of the same image and the second layer from the NL-MEANs (a digital image denoising method, based on a non local averaging of all present pixels in an image) output. The RGB image is obtained from adding the two layers to the original X-ray image [11].

4.4.2 Guangzhou Women and Children's Medical Center Pediatric Database

The only preprocessing done on the Guangzhou Women and Children's Medical Center Pediatric Database was resizing all images to 320×320.

Chapter 5

Experiments, results and discussions

Leveraging unlabelled data to pretrain models for classification and detection could potentially have an impact on medical imaging similar to ImageNet's impact on computer vision in general. With the experiments we attempt to answer the following questions:

- Can we use an autoencoder for feature learning from unlabelled x-ray images?
- Can we apply these learned features to an x-ray classification task with a small dataset?
- What characteristic of the unlabelled dataset would affect the performance of the proposed method?

In this chapter we run experiments to evaluate if using unlabelled medical imaging data to train autoencoder for reconstruction, then transferring the learned feature maps to the target task, improves performance and performs better than transfer learning with natural image datasets (e.g. ImageNet). We also investigate the impact of the size of the unlabelled source dataset, and other characteristics (like whether the unlabelled medical dataset is broad and diverse, homogeneous, or exactly the same as the target dataset) on the performance of the proposed method.

The chapter is structured as follows. We train the classification model from scratch with the target dataset as our baseline, and to estimate the effectiveness of the proposed method we compare the results between the baseline and the proposed method in Section 5.1. We explore different sizes and characteristics of source datasets in Section 5.2 and 5.3. We also compare our method with ImageNet models on the target dataset in Section 5.4.

5.1 Main experiment: proposed method

The main question we want to explore is whether an autoencoder can learn features from an unlabelled image dataset and if these feature can be useful to pretrain a classification CNN for pneumonia detection. In this section we train the CBR-LargeT model from scratch on the target dataset as the baseline in Section 5.1.1. To evaluate the performance of the proposed method we train an autoencoder for reconstruction on the source dataset for feature learning in Section 5.1.2. After that we transfer the kernels/weights from the autoencoder to the CBR-LargeT model and then train on the target dataset for classification in Section 5.1.3.

5.1.1 Baseline with CBR-LargeT model for target task

For the target task we train the CBR-LargeT model with the GWCM dataset images resized to 320×320. We split the data for training and validation, with 5232 x-ray images (where 1349 are normal and 3883 have been diagnosed as having pneumonia) and 624 x-ray images respectively. A hyperparameter grid search [53] of the batch size and learning rate was done, other parameters were chosen based on standards in literature. We train with a learning rate of 0.01, a decay parameter of 1e-6, a momentum parameter of 0.9, categorical cross entropy loss, and a batch size of 32. After training for 200 epochs the model achieved an accuracy of 86.859% (with a precision, recall, and F1 score of 87%).

Table 5.1: CBR-LargeT baseline performance.

	accuracy	precision	recall	f1-score
CBR-LargeT (baseline)	86.859%	87%	87%	87%

5.1.2 Unsupervised reconstruction task with ImageCLEFmed 2009

To simulate an unlabelled dataset we use the ImageCLEFmed 2009 with the labels removed. The dataset is partitioned into two sets: the training set which has 12677 images and the test set which has 1733 images. For data preprocessing we simply resize the images from 512×512 to 320×320. The autoencoder is trained on the 320×320 images, with a learning rate of 0.01, a decay parameter of 1e-6, a momentum parameter of 0.9, and a batch size of 32. The SGD optimizer is used.

After training, the reconstruction loss (which is an estimate of the uncertainty between input and the predicted output of the reconstruction) of the autoencoder is 0.32572. The reconstruction results can be seen in Figure 5.1. The top row consists of the original images from the ImageCLEFmed 2009 dataset, and the bottom row are the images reconstructed by the autoencoder. The autoencoder can differentiate between the background and the foreground but does not seem to produce any detailed information about the foreground.

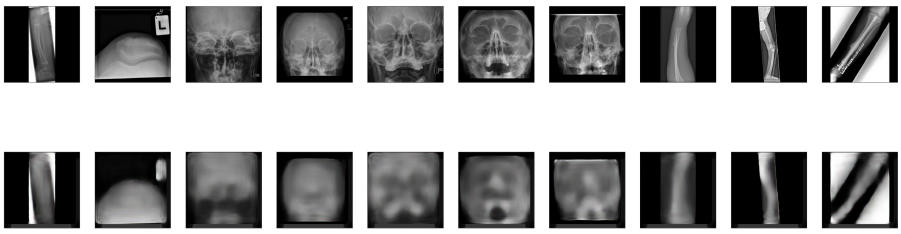

Figure 5.1: ImageCLEFmed 2009 reconstruction output at 320×320 image size, with the original images at the top and the reconstructions at the bottom.

The experiment was repeated with the images in the training set resized to 512×512. The reconstruction results can be improved significantly by increasing the image size to 512×512 (see Figure 5.2), but this would significantly reduce the maximum size we can have for the reconstruction task due to the limited computational resources. Therefore, we continue with the autoencoder trained with 320×320 images. Even though the autoencoder does not seem to perform well at reconstruction we assume useful features are learnt by the encoder for transfer onto a classification task. An autoencoder is has two main components (encoder and decoder) but we are only transferring representations learned by the encoder. Therefore, it may difficult say if encoder has learned significant features from looking at the performance of the full autoencoder on a reconstruction task.

5.1.3 Target Task: Transferred kernels and weights from ImageCLEFmed 2009 reconstruction

Semi-supervised transfer learning is applied to the CBR-LargeT model for the target dataset. As described in Section 4.1, the kernels and weights from the encoder part of the autoencoder (trained in Section 5.1.2) are transferred to the CBR-LargeT model. For the training, hyperparameters are kept the same as for the baseline training described in Section 5.1.1. After training for 200 epochs the model achieved an accuracy of 91.026% (with a precision,

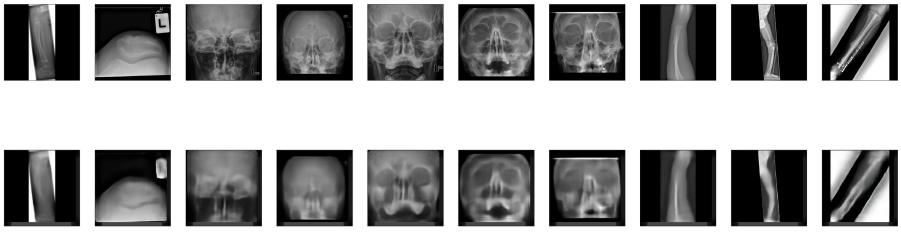

Figure 5.2: ImageCLEFmed 2009 reconstruction output at 512×512 image size, with the original images at the top and the reconstructions at the bottom.

recall, and F1 score of 91%).

Table 5.2: CBR-LargeT performance with pre-trained with features learned from the ImageCLEFmed 2009 reconstruction task.

	accuracy	precision	recall	f1-score
CBR-LargeT (12.6k)	91.026%	91%	91%	91%

5.1.4 Results

Our proposed method improves the baseline performance by 4.167% in accuracy and improves precision, recall and F1 score by 4%. Transferring from a relatively small unlabelled dataset of 12667 images had a significant increase in accuracy. Traditional transfer learning has significantly larger datasets (i.e. ImageNet with over 14 million images).

In the following sections we try to answer a few questions about the proposed method. Does increasing the source dataset improve its performance? Does the source dataset have to be broad and diverse, or can it be homogeneous?

5.2 Increasing unlabelled data

In the previous section we demonstrate that features learned by an autoencoder from unlabelled x-ray images can be transferred to a CNN for pneumonia detection to produce good results (see Section 5.1.3). In this section we explore the effects of increasing the size of the source dataset on the performance of the proposed method.

Transfer learning traditionally has large scale source datasets with millions of images. Usually, the larger the source dataset the better the performance. Our proposed method performs well with a source dataset that only has 12677 images. In this section we increase the source dataset to 17893 images, repeat the experiments in Section 5.1, and evaluate the results to see if the expanded source dataset produces a better performance.

5.2.1 Unsupervised reconstruction task with ImageCLEFmed 2009 + GWCM

We increase the size of the source dataset for reconstruction by combining the images from the ImageCLEFmed 2009 and GWCM datasets. The resulting dataset has 17893 images in the training set and 2357 images in the test set. All these images are resized to 320×320. For the training, hyperparameters are kept the same as for the initial reconstruction task in Section 5.1.2.

The reconstruction loss achieved by the autoencoder on this dataset is 0.39706. The autoencoder's reconstruction loss is worse than in Section 5.1.2, and it reconstructs images that are comparable and do not seem different to the naked eye. The reconstruction results can be seen in Figure 5.3.

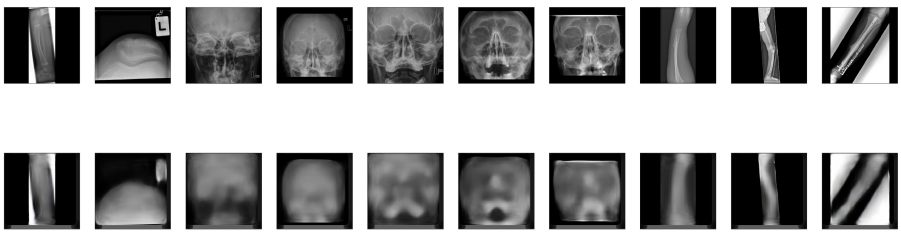

Figure 5.3: Reconstruction outputs with 320×320 images when the autoencoder is trained on unlabelled x-ray images from a combination of the ImageCLEFmed 2009 and GWCM datasets (17893 images in the training set and 2357 in the test set). The original images are at the top and the reconstructions are at the bottom.

5.2.2 Target Task: Transferred kernels and weights from ImageCLEFmed 2009 + GWCM reconstruction

Training for the target task with the weights from the ImageCLEFmed 2009 + GWCM reconstruction task was done with the same hyperparameters as for the baseline training described in Section 5.1.1. After training for 200 epochs the model achieved an accuracy of 92.147% (with a precision, recall, and F1 score of 92%).

Table 5.3: CBR-LargeT performance on the target task when pre-trained with features learned from the ImageCLEFmed 2009+ GWCM reconstruction task in Section 5.2.1.

	accuracy	precision	recall	f1-score
CBR-LargeT (17.8k)	92.147%	92%	92%	92%

5.2.3 Results

Increasing the source dataset from 12677 image to 17893 images improved our method's accuracy by 1.121%. This resulted in an overall 5.2884% accuracy increase (5% increase in recall, precision and F1 score) from the baseline.

From these results we can assume by extrapolation that increasing the unlabelled source dataset to a million or even hundreds of thousands would produce significantly better results.

5.3 Broad and diverse unlabelled dataset

What to transfer is an important issue in transfer learning. Choosing an appropriate source dataset needs to be deliberated. Should the source dataset be broad and diverse, homogeneous, or exactly the same as the target dataset to produce the best results?

We run the following experiments to investigate significant characteristics required from the source dataset. We use the same dataset for the source and target task in Section 5.3.1. In Section 5.3.2 use a source dataset that is very similar to the target dataset but significantly larger to see the effects of the dataset size in this regard.

5.3.1 Same Source and target dataset

In this section we use the same dataset to train the autoencoder for reconstruction as the target task (pneumonia detection with the GWCM dataset) to see if representations learned by the autoencoder will provide better results. We reduce the size of the source dataset for reconstruction by training the autoencoder on the GWCM dataset. The resulting dataset has 5216 images in the training set and 624 images in the test set. All these images are resized to 320×320. For the training, hyperparameters are kept the same as for the initial reconstruction task in Section 5.1.2.

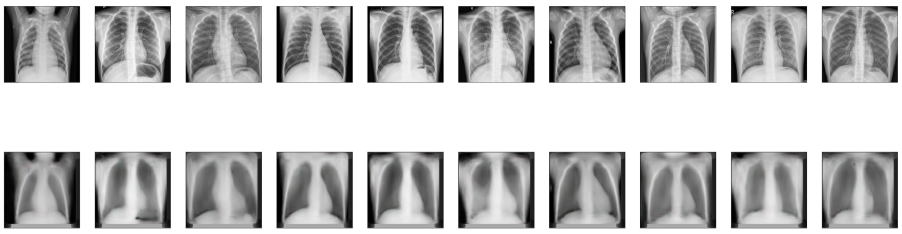

Figure 5.4: GWCM reconstruction output at 320×320 image size, with the original images at the top and the reconstructions at the bottom.

Training for the target task with the weights from the GWCM reconstruction task was done with the same hyperparameters as for the baseline training described in Section 5.1.1. After training for 200 epochs the model achieved an accuracy of 88.141% (with a precision, recall, and F1 score of 88%).

Table 5.4: CBR-LargeT performance with the autoencoder pre-trained on the GWCM dataset which is the same as the target task.

	accuracy	precision	recall	f1-score
CBR-LargeT (5.2k)	88.141%	88%	88%	88%

Using the same dataset for the source and target task does not produce better results. This method resulted in an overall 1.282% accuracy (1% increase in recall, precision and F1 score) increase from the baseline, but performed worse than having the source dataset as ImageCLEFmed 2009 or ImageCLEFmed 2009 + GWCM (in Section 5.1.3 and 5.2.2 respectively). The size of the training dataset for the reconstruction may be the more significant factor for producing better results from transferring the weights. Transfer learning generally needs a large amount of data. To test this, we evaluate our method on a larger pneumonia dataset.

5.3.2 RSNA

In this section we use a similar but significantly larger dataset (RSNA). We train the autoencoder for reconstruction as the target task (pneumonia detection with the GWCM dataset) to see if the representation learned by the autoencoder from a larger dataset will provide better results.

We use a large source dataset for reconstruction namely the RSNA dataset. This dataset has 23115 images in the training set. All these images are resized to 320×320. For the training, hyperparameters are kept the same as for the initial reconstruction task in Section 5.1.2.

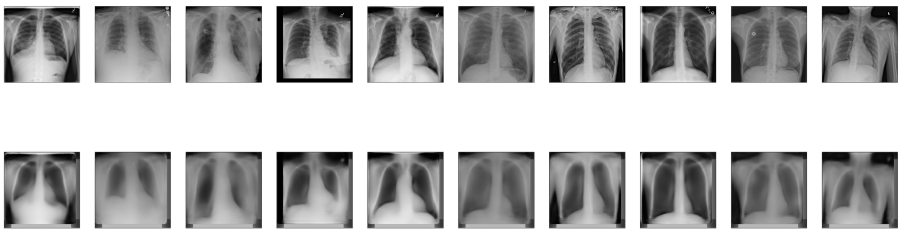

Figure 5.5: RSNA reconstruction output at 320×320 image size, with the original images at the top and the reconstructions at the bottom.

Training for the target task with the weights from the RSNA reconstruction task was

done with the same hyperparameter as for the baseline training described Section 5.1.1. After training for 200 epochs the model achieved an accuracy of 89.904% (with a precision, recall, and F1 score of 90%).

Table 5.5: CBR-LargeT performance with features learned from the RSNA reconstruction task.

	accuracy	precision	recall	f1-score
CBR-LargeT (23.1k)	89.904%	90%	90%	90%

Using a source dataset that is similar but significantly larger than the target dataset performs slightly better than using the same dataset as source and target. This approach has a 1.763% increase in accuracy compared to the same dataset approach and a 3.045% improvement compared to the baseline. Even though the source dataset of this approach is larger than the first dataset in the previous sections (ImageCLEFmed 2009 with 12677 image in Section 5.1.3 and ImageCLEFmed 2009 + GWCM with 17893 images in Section 5.2.2), it does not have better performance. Therefore, the size of the source dataset is not the only contributing factor to the performance of the proposed method. Having a source dataset that is too similar to the target dataset may not perform as well as the approaches in Sections 5.1.3 and 5.2.2 due to over-fitting and/or because the proposed method used the concepts of transfer learning which require a sufficiently broad and diverse large source dataset.

5.4 ImageNet model: Target task

For the standard ImageNet architectures, we evaluate ResNet50 and Inception-v3, which have both been used extensively in medical transfer learning applications. We use ResNet50 and Inception-v3 pre-trained on ImageNet and retrain on the target dataset. For the training, hyperparameters are kept the same as for the baseline training described in Section 5.1.1. Table 5.6 shows our proposed method's performance compared to two standard ImageNet models on the GWCM dataset for pneumonia detection.

Table 5.6: Model performances for our proposed method and standard ImageNet architectures.

	accuracy	precision	recall	f1-score
CBR-LargeT (baseline)	86.859%	87%	87%	87%
CBR-LargeT (12.6k)	91.026%	91%	91%	91%
CBR-LargeT (17.8k)	92.147%	92%	92%	92%
Inception-v3	89.423%	90%	89%	89%
ResNet50	85.417%	88%	85%	84%

The baseline is comparable to ImageNet's ResNet50 performance. The proposed method produces a performance improvement of 4.167% and 5.2884% with 12667 and 17865 images respectively from the baseline. Inception-V3 produces a performance improvement of only 2.564% from the baseline. The proposed method performs slightly better than state of the art ImageNet models even though ImageNet is 70 to 90 times larger than the source datasets used in our experiments.

Chapter 6

Conclusion

In this work we propose a semi-supervised transfer learning approach for training deep learning models for medical images, in response to the problem of insufficient labelled medical imaging data. The main idea behind the proposed method is to leverage unlabelled medical image datasets to improve model accuracy for the target task by transferring feature maps learned from an unsupervised task to the supervised target task. We leverage unlabelled data by transferring weights/kernels learned by an autoencoder (specifically the encoder part) during an unsupervised reconstruction task (from a large collection of unlabelled medical images) to a classification task. The proposed approach can be used on any medical image, but we focus on x-ray images in this work.

We investigate whether the proposed method improves the performance of the model compared to the baseline (i.e. to train the model from scratch) and state of the art ImageNet models. We also investigate the impact of the size of the unlabelled source dataset, and other characteristics (like whether the unlabelled medical dataset is broad and diverse, homogeneous, or exactly the same as the target dataset) on the performance of the proposed method.

Our proposed method improves the baseline performance by 4.167% in accuracy and improves precision, recall and F1 scores by 4%. This confirms that the proposed method successfully leverages learning from unlabelled data. Our exploration of the impact of size of the unlabelled source dataset demonstrated that increasing the source dataset from 12677 images to 17893 images improved our method's accuracy by 1.121%. This shows that the size of the source dataset has a positive impact on the performance of our proposed approach, so increasing the size of the source dataset increases the performance.

In investigating significant characteristics required from the source dataset we discover that the size of the unlabelled source dataset is not the only significant factor to the performance of the proposed approach. We find that using the same dataset for the source and target task does not produce better results. Even though the source dataset of this approach is larger than the first dataset in the previous sections (ImageCLEFmed 2009 with 12677 images in Section 5.1.3, and ImageCLEFmed 2009 + GWCM with 17893 images in Section 5.2.2), it does not have better performance. Therefore, the size of the source dataset is not the only contributing factor to the performance of the proposed method. Having a source dataset that is too similar to the target dataset may not perform as well as the approaches in Sections 5.1.3 and 5.2.2 due to over-fitting and/or because the proposed method used the concepts of transfer learning which require a sufficiently broad and diverse large source dataset.

We also compare our method with ImageNet models on a target dataset. For the standard ImageNet architectures, we evaluate ResNet50 and Inception-v3, which have both been used extensively in medical transfer learning applications. Our proposed method outperforms both standard ImageNet models on the target task. Other work on target dataset include Walia et al. [54] with a VGG16 and no data augmentation they achieved an accuracy of 80% (68% precision, 80% recall and 73.515 F1 score) and with a VGG19 (with data augmentation) achieved an accuracy of 86% (71% precision, 87% recall and 78.518 F1 score). Karan Jakhar et al.[55] used a Deep CNN and achieved and accuracy of 84% using tenfold cross-validation technique to validate the model. We achieve results that a better that these works.

One central downside of supervised transfer learning is that a large quantity of expensive manually-supervised training data is required. Hence transfer learning from natural image datasets has become a standard even for medical imaging. However, this approach has been shown to be ineffective due to the significant differences between medical images and natural images. Developing a large-scale medical imaging dataset for transfer learning would be too expensive, therefore the possibility of using large amounts of unlabelled data for feature learning is very attractive. We show that autoencoders can be used to effectively transfer knowledge learned from an unsupervised learning task (reconstruction with unlabelled data) to a classification task to improve its performance. These findings could potentially be extended to creating an arsenal of pre-trained architectures for medical imaging tasks to have an impact to medical imaging similar to ImageNet's impact on computer vision in general.

6.1 Future Work

Experiments done in this work were limited by computing resources and access to datasets. Therefore, there is significant potential to improve and extend this work.

6.1.1 More general

The results of this work focus on a pneumonia x-ray dataset, with two classes and one anatomical structure (chest). The results are therefore not sufficiently general, and more domains need to be explored. For example, experiments could be done with:

- a target dataset with multiple pathologies
- a target dataset with multiple anatomical structures (hand, head, knee etc.)
- a target dataset with other modalities (MRI, CT, etc.).

6.1.2 Source dataset characteristic

A more in-depth exploration of the source dataset characteristic to optimize the proposed method is needed.

Bibliography

[1] A. Misra, "Making art with your webcam," Mar 2019. [Online]. Available: https://towardsdatascience.com/making-art-with-your-webcam-ac6d0f5504f4

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[3] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into Imaging*, vol. 9, no. 4, pp. 611–629, Aug 2018. [Online]. Available: https://doi.org/10.1007/s13244-018-0639-9

[4] S. Saha, "A comprehensive guide to convolutional neural networks," Dec 2018. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[5] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *J. Mach. Learn. Res.*, vol. 10, p. 1633–1685, dec 2009.

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[8] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2018.

[9] F. Pelka, Nensa, "Annotation of enhanced radiographs for medical image retrieval with deep convolutional neural networks," 2018.

[10] "Chest x-ray images (pediatric pneumonia)." [Online]. Available: https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia/home

[11] X. Nkwentsha, A. Hounkanrin, and F. Nicolls, "Automatic classification of medical x-ray images with convolutional neural networks," in *2020 International SAUPEC/RobMech/PRASA Conference*, 2020, pp. 1–4.

[12] D. Shen, G. Wu, and H.-I. Suk, "Deep learning in medical image analysis," *Annual Review of Biomedical Engineering*, vol. 19, no. 1, pp. 221–248, 2017.

[13] D. Sultana, Sufian, "Advancements in image classification using convolutional neural network," 2019.

[14] X. W. Zhao, Zheng, "Object detection with deep learning: A review," 2019.

[15] Y. Liu, Deng, "Recent progress in semantic image segmentation," 2018.

[16] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60 – 88, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1361841517301135

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," 2015.

[18] M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio, "Transfusion: Understanding transfer learning for medical imaging," 2019.

[19] Q. H. Nguyen, B. P. Nguyen, S. D. Dao, B. Unnikrishnan, R. Dhingra, S. R. Ravichandran, S. Satpathy, P. N. Raja, and M. C. H. Chua, "Deep learning models for tuberculosis detection from chest x-ray images," in *2019 26th International Conference on Telecommunications (ICT)*, 2019, pp. 381–385.

[20] R. Roomaney, V. Pillay-van Wyk, O. Awotiwon, A. Dhansay, J. Joubert, M. Nglazi, E. Nicol, and D. Bradshaw, "Epidemiology of lower respiratory infection and pneumonia in South Africa (1997-2015): a systematic review protocol," *BMJ open*, vol. 6(9), 2016.

[21] W. H. Organization, "Standardization of interpretation of chest radiographs for the diagnosis of pneumonia in children," 2001.

[22] T. Rahman, M. E. H. Chowdhury, A. Khandakar, K. R. Islam, K. F. Islam, Z. B. Mahbub, M. A. Kadir, and S. Kashem, "Transfer learning with deep convolutional neural network for pneumonia detection using chest x-ray," *Applied Sciences*, vol. 10, no. 9, p. 3233, May 2020. [Online]. Available: http://dx.doi.org/10.3390/app10093233

[23] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436—444, May 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[24] A. Khan, A. Sohail, U. Zahoora, and Q. A. Saeed, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, p. 5455–5516, 2020.

[25] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, 2012. [Online]. Available: http://arxiv.org/abs/1207.0580

[26] B. J. Kingma DP, "Adam: a method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/pdf/1412.6980.pdf.

[27] R. S, "An overview of gradient descent optimization algorithms," 2014. [Online]. Available: https://arxiv.org/pdf/1609.04747.pdf.

[28] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Networks*, vol. 12, no. 1, pp. 145–151, 1999. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608098001166

[29] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60 – 88, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1361841517301135

[30] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 2016, pp. 265–283. [Online]. Available: https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

[31] R. Collobert, S. Bengio, and J. Marithoz, "Torch: A modular machine learning software library," 11 2002.

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *NeurIPS*, 2019.

[33] Theano Development Team, "Theano: A python framework for fast computation of mathematical expressions," *CoRR*, vol. abs/1605.02688, 2016. [Online]. Available: http://arxiv.org/abs/1605.02688

[34] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *CoRR*, vol. abs/1408.5093, 2014. [Online]. Available: http://arxiv.org/abs/1408.5093

[35] F. Chollet, "Keras," https://github.com/fchollet/keras, 2015.

[36] A. Vedaldi and K. Lenc, "Matconvnet - convolutional neural networks for MATLAB," *CoRR*, vol. abs/1412.4564, 2014. [Online]. Available: http://arxiv.org/abs/1412.4564

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[38] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[39] M. A. Morid, A. Borjali, and G. Del Fiol, "A scoping review of transfer learning research on medical image analysis using imagenet," *Computers in Biology and Medicine*, vol. 128, p. 104115, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010482520304467

[40] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[41] D. Kim and T. MacKinnon, "Artificial intelligence in fracture detection: transfer learning from deep convolutional neural networks," *Clinical Radiology*, vol. 73, no. 5, pp. 439–445, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0009926017305354

[42] J. Yu, S. Yu, B. Erdal, M. Demirer, V. Gupta, M. Bigelow, A. Salvador, T. Rink, S. Lenobel, L. Prevedello, and R. White, "Detection and localisation of hip fractures on anteroposterior radiographs with artificial intelligence: proof of concept," *Clinical Radiology*, vol. 75, no. 3, pp. 237.e1–237.e9, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0009926019306403

[43] J.-H. Lee, D.-H. Kim, S.-N. Jeong, and S.-H. Choi, "Detection and diagnosis of dental caries using a deep learning-based convolutional neural network algorithm," *Journal of Dentistry*, vol. 77, pp. 106–111, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0300571218302252

[44] M. Ahsan, R. Gomes, and A. Denton, "Application of a convolutional neural network using transfer learning for tuberculosis detection," in *2019 IEEE International Conference on Electro Information Technology (EIT)*, 2019, pp. 427–433.

[45] S. A. Prajapati, R. Nagaraj, and S. Mitra, "Classification of dental diseases using cnn and transfer learning," in *2017 5th International Symposium on Computational and Business Intelligence (ISCBI)*, 2017, pp. 70–74.

[46] W. Poedjiastoeti and S. Suebnukarn, "Application of convolutional neural network in the diagnosis of jaw tumors," *Healthc Inform Res. 2018;24(3):236-241.*, vol. 24.3, pp. 236–241, 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6085208/

[47] M. Tschannen, O. Bachem, and M. Lucic, "Recent advances in autoencoder-based representation learning," 2018.

[48] F. Li, H. Qiao, B. Zhang, and X. Xi, "Discriminatively boosted image clustering with fully convolutional auto-encoders," *CoRR*, vol. abs/1703.07980, 2017. [Online]. Available: http://arxiv.org/abs/1703.07980

[49] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 1753–1759. [Online]. Available: https://doi.org/10.24963/ijcai.2017/243

[50] L. Hou, V. Nguyen, A. Kanevsky, D. Samaras, T. Kurc, T. Zhao, R. Gupta, Y. Gao, W. Chen, D. Foran, and J. Saltz, "Sparse autoencoder for unsupervised nucleus detection and representation in histopathology images," *Pattern Recognition*, vol. 86, 09 2018.

[51] E. Hosseini asl, R. Keynto, and A. El-Baz, "Alzheimer's disease diagnostics by adaptation of 3d convolutional network," 07 2016.

[52] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, pp. 1452–1464, 2018.

[53] J. Brownlee, "How to grid search hyperparameters for deep learning models in python with keras," Aug 2020. [Online]. Available: https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/

[54] I. S. Walia, M. Srivastava, D. Kumar, M. Rani, P. Muthreja, and G. Mohadikar, "Pneumonia detection using depth-wise convolutional neural network (dw-cnn)," *EAI Endorsed Transactions on Pervasive Health and Technology*, vol. 81, no. 23, 9 2020.

[55] K. Jakhar and N. Hooda, "Big data deep learning framework using keras: A case study of pneumonia prediction," in *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, 2018, pp. 1–5.