QUALITY ENHANCEMENT OF COMPUTED TOMOGRAPHY IMAGES OF POROUS MEDIA USING CONVOLUTIONAL NEURAL NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ERTUĞRUL UMUT YILDIRIM

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
SCIENTIFIC COMPUTING

FEBRUARY 2022

Approval of the thesis:

QUALITY ENHANCEMENT OF COMPUTED TOMOGRAPHY IMAGES OF POROUS MEDIA USING CONVOLUTIONAL NEURAL NETWORKS

submitted by **ERTUĞRUL UMUT YILDIRIM** in partial fulfillment of the requirements for the degree of **Master of Science in Scientific Computing Department, Middle East Technical University** by,

Prof. Dr. A. Sevtap Kestel
Dean, Graduate School of **Applied Mathematics** ______________

Prof. Dr. A. Sevtap Kestel
Head of Department, **Scientific Computing** ______________

Prof. Dr. Ömür Uğur
Supervisor, **Scientific Computing, IAM, METU** ______________

Assist. Prof. Dr. Guenther Glatz
Co-supervisor, **Petroleum Engineering Department, KFUPM** ______________

Examining Committee Members:

Prof. Dr. Ayhan Aydın
Mathematics Department, Atılım University ______________

Prof. Dr. Ömür Uğur
Scientific Computing, IAM, METU ______________

Assist. Prof. Dr. Önder Türk
Scientific Computing, IAM, METU ______________

Date: ______________

ABSTRACT

QUALITY ENHANCEMENT OF COMPUTED TOMOGRAPHY IMAGES OF POROUS MEDIA USING CONVOLUTIONAL NEURAL NETWORKS

Yıldırım, Ertuğrul Umut
M.S., Department of Scientific Computing
Supervisor : Prof. Dr. Ömür Uğur
Co-Supervisor : Assist. Prof. Dr. Guenther Glatz

February 2022, 43 pages

Computed tomography has been widely used in clinical and industrial applications as a non-destructive visualization technology. The quality of computed tomography scans has a strong effect on the accuracy of the estimated physical properties of the investigated sample. X-ray exposure time is a crucial factor for scan quality. Ideally, long exposure time scans, yielding large signal-to-noise ratios, are available if physical properties are to be delineated. However, especially in micro-computed tomography applications, long exposure times constitute a problem for monitoring some physical processes that are happening quickly. To alleviate this problem, this thesis proposes a convolutional neural network approach for scan quality enhancement allowing for a reduction in X-ray exposure time while improving signal-to-noise ratio of the scanned image simultaneously. Moreover, the impact of using different loss functions, namely the mean squared error and the structural similarity index measure, on the performance of the network is analyzed. Both the visual and quantitative assessments show that the trained network greatly improves the quality of low-dose scans.

Keywords: convolutional neural networks, computed tomography, denoising

ÖZ

EVRİŞİMLİ SİNİR AĞLARI KULLANARAK GÖZENEKLİ ORTAMIN BİLGİSAYARLI TOMOGRAFİ GÖRÜNTÜLERİNİN KALİTESİNİN İYİLEŞTİRİLMESİ

Yıldırım, Ertuğrul Umut
Yüksek Lisans, Bilimsel Hesaplama Bölümü
Tez Yöneticisi : Prof. Dr. Ömür Uğur
Ortak Tez Yöneticisi : Yrd. Doç. Dr. Guenther Glatz

Şubat 2022, 43 sayfa

Bilgisayarlı tomografi, tahribatsız bir görselleştirme teknolojisi olarak klinik ve endüstriyel uygulamalarda yaygın olarak kullanılmaktadır. Bilgisayarlı tomografi taramalarının kalitesi, incelenen örneğin tahmini fiziksel özelliklerinin doğruluğu üzerinde güçlü bir etkiye sahiptir. X-ışınına maruz kalma süresi, tarama kalitesi için çok önemli bir faktördür. İdeal olarak, fiziksel özellikler tanımlanacaksa, büyük sinyal-gürültü oranları sağlayan uzun maruz kalma süresi taramaları kullanışlıdır. Ancak özellikle mikro bilgisayarlı tomografi uygulamalarında uzun pozlama süreleri hızlı gerçekleşen bazı fiziksel süreçlerin izlenmesinde sorun teşkil etmektedir. Bu sorunu hafifletmek için, bu tez, taranan görüntünün sinyal-gürültü oranını aynı anda iyileştirirken, X-ışını maruz kalma süresinde bir azalmaya izin veren tarama kalitesi geliştirmesi için evrişimli bir sinir ağı yaklaşımı önermektedir. Ayrıca, ortalama karesel hata ve yapısal benzerlik indeksi ölçüsü gibi farklı kayıp fonksiyonlarının kullanılmasının ağın performansı üzerindeki etkisi analiz edilmiştir. Hem görsel hem de nicel değerlendirmeler, eğitimli ağın düşük dozlu taramaların kalitesini büyük ölçüde iyileştirdiğini göstermektedir.

Anahtar Kelimeler: evrişimli sinir ağları, bilgisayarlı tomografi, gürültü giderme

To my family

ACKNOWLEDGMENTS

I would like to acknowledge my appreciation to my thesis supervisor Prof. Dr. Ömür Uğur for his assistance and valuable comments throughout the production of this thesis. I would also like to express my sincere gratitude to my co-supervisor Assist. Prof. Dr. Guenther Glatz for his precious contributions that helped me to complete this work. Without them, I would not have been able to finalize my thesis.

I would also like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for supporting me financially with National Masters Degree Scholarship Program during my graduate studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

CT	Computed Tomography
DRP	Digital Rock Physics
Micro-CT	Micro-computed Tomography
DCNN	Deep Convolutional Neural Network
LDCT	Low-dose Computed Tomography
MSE	Mean Squared Error
SSIM	Structural Similarity Index Measure
PSNR	Peak Signal-to-Noise Ratio
ANN	Artificial Neural Network
MLP	Multilayer Perceptron
BGD	Batch Gradient Descent
SGD	Stochastic Gradient Descent
MBD	Mini-batch Gradient Descent
CNN	Convolutional Neural Network
FCN	Fully Convolutional Neural Network
MSSIM	Mean Structural Similarity Index Measure
TPU	Tensor Processing Unit

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

X-ray computed tomography (CT) is a widely used imaging technique in the health-care domain, largely because of its ability to provide a detailed insight into the human body noninvasively. Computed tomography technology is not only used for diagnostic purposes in radiology but also has been utilized in many industrial applications. One of the examples of CT application in industry is the detection of flaws (e.g. voids, cracks) in materials [10]. The technology has also been adopted for measurement of components for a manufacturing process and quality testing of complex assemblies [29]. Reverse engineering is another important field in which CT scanning has found applications [2].

Being a non-destructive technology, X-ray CT has also been widely used in various geological fields. Application areas range from rock and soil mechanics to petroleum geology [22]. Imaging 3D geometry of the mineral phase and the pore-space of reservoir rock at the pore scale is very useful in revealing the internal structure of the rock. Especially in digital rock physics (DRP) applications, high resolution digital images of porous media enable us to quantify some of the important rock properties such as porosity, permeability and resistivity with the help of numerical simulations.

Figure 1.1 shows a typical digital rock workflow. There are several main steps in the process. The first one is collecting a series of projections at different angles by scanning of a specimen using a CT technology. Then, a reconstruction algorithm is implemented to generate tomographic images, which are then used to create a 3D

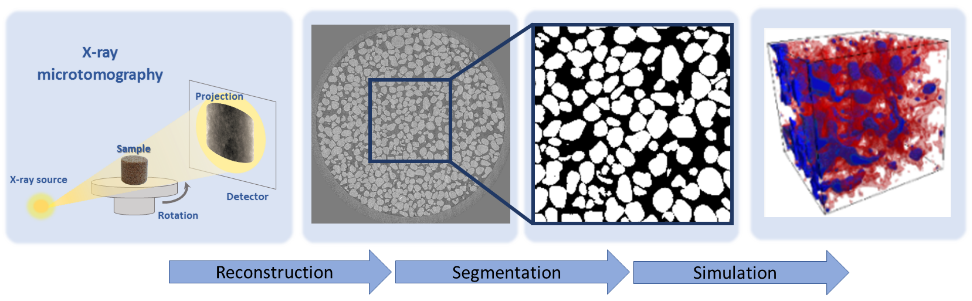


Figure 1.1: A typical digital rock workflow [27].

image of the sample. The next step is the segmentation of the reconstructed gray-scale images to distinguish between the pore and solid grain space. Lastly, numerical simulations are performed on the digital image of rock to identify the properties of interest. Numerical simulation of fluid flow based on high-resolution tomographic images of rock and prediction of permeability are presented in [23, 4].

The quality, resolution, and size of CT scans are of utmost importance when it comes to measuring the upscaled rock properties accurately. For higher resolution scans, X-ray micro-computed tomography (micro-CT) has been extensively benefited in the area of DRP [1]. Contrary to conventional CT, micro-CT can provide images with a voxel resolution as small as a few micrometers in size. There is a trade-off between the resolution of the image and the physical size of the sample being investigated. The higher the resolution of the scan, the smaller the size of the core sample [3]. Finer resolution helps detect some details such as grain shapes and roughness, whereas the size of the sample should be large enough to be representative for a particular rock property.

Image quality is another crucial factor that affects the accuracy of the properties estimated from CT data. There are many different artifacts that can deteriorate CT image quality. These artifacts include noise, beam hardening, motion blur, ring and metal artifacts, to name a few [5].

Increasing X-ray exposure time at each projection view would lead to longer imaging time and higher radiation dose, thus giving rise to better image quality [14]. In medical domain, minimizing the associated radiation dose is especially important as it may have a negative impact on patient's health. When it comes to geoscience applications, however, the radiation dose is not an issue. Nevertheless, high-density components of geomaterials cause more attenuation of X-rays, so higher exposure time is required to obtain high-quality images. This necessitates prolonged acquisition times causing a serious problem when certain processes happening too fast are to be monitored. For example, in high-temperature experiments, it is hard to capture the dynamics of certain reactive processes because of the fact that increasing the temperature results in an exponential rise in the rate of the chemical reaction. [12]. Thus, these types of experiments require low exposure times for monitoring the process effectively. Especially, in micro-CT technologies, the restriction of prolonged acquisition times becomes more apparent as the time for completion of the scan may take several hours. Nevertheless, lower exposure of X-ray flux makes the reconstructed images noisier, which degrades the signal-to-noise ratio [7]. Therefore, one needs to apply denoising methods to improve the quality of such CT images. Since noise reduction causes the significant features of the image to get lost, preserving these features such as edges, corners, and other sharp structures is extremely important while denoising the image [11].

Various approaches have been proposed in the literature to improve the quality of low-dose scans. One of those approaches is based on sinogram filtering and, therefore, requires access to the projection data. In this method, noise removal filters are applied to raw data before a reconstruction algorithm such as filtered back-projection [18, 21]. Usually, it is difficult to obtain the projection data as most of the vendors do not provide them. Another common way to deal with low-dose CT (LDCT) images is using iterative reconstruction techniques [30]. These techniques require some prior information and cost high amount of computation.

Taking into consideration the restrictions caused by the aforementioned methods, post processing techniques which can be directly applied to low quality images, can be considered as a useful alternative. Among them, deep neural network based approaches have been used commonly in the area of CT image denosing. In the medical domain, the application of deep convolutional neural networks (DCNNs) has shown a great success in reducing the noise level in LDCT images, thus minimizing patient radiation exposure while preserving image quality and diagnostic power [31, 7, 8].

1.2 Objective and Scope of the Thesis

The aim of this thesis is to apply convolutional neural networks to computed tomography scans of porous rock samples to increase the quality of reduced exposure images by reducing the associated noise. In other words, training the proposed network by mapping low-quality images to higher-quality images, the network is expected to act as a denosing filter once applied to a given low-dose scan.

There are several advantages of reducing the scanning time. First, it causes an increase in temporal resolution; meaning that even though the exposure time is lessened, higher-quality images can be obtained. Thus, capturing the dynamics of certain type of experiments can be possible due to the reduced scan time. Secondly, image quality has a strong effect on the accuracy of the estimated rock properties in digital rock physics applications. With the help of the trained network, noise reduction is accomplished in the low-quality images. This will also help in the image processing steps like resolution enhancement. Third, lower scan times have a positive impact on the lifetime of the X-ray tube or the filament.

For this work, actual micro-CT data was generated by a *FEI Heliscan microCT* operating with a cone-beam. The machine was configured to perform 1800 projections per revolution. Tube voltage and current are adjusted as 85 kV and 72 mA, respectively. Two datasets were acquired by altering the exposure time for each projection. The first one is composed of the low-quality, low-dose scans which were produced by setting the exposure time as 0.5 s. The second one is composed of the high-quality, higher-dose scans resulting from 1.4 s exposure time for each projection. Subsequently, a U-net based DCNN was trained using two different loss functions namely, the mean squared error (MSE) and the structural similarity index measure (SSIM). The results were assessed both quantitatively and qualitatively. For the quantitative analysis, two commonly used image metrics, peak signal-to-noise ratio (PSNR) and the SSIM were utilized. Lastly, artificial datasets were created based on the predictions, which are deemed as the ground truth, from the previously trained neural network such that they mimic both low-exposure and high-exposure scans. These datasets were used to validate the prediction power of the neural network at denoising the low-quality images.

1.3 Outline of the Thesis

This thesis is divided into four chapters. In Chapter 1, the problem is defined and the motivation behind this work is given. Later, the aim of the thesis is explained. In Chapter 2, the theory of deep neural networks is outlined with a special emphasis on convolutional neural networks, which play a major role in this study. In Chapter 3, the proposed network and the datasets on which the network is trained are explained in details. Additionally, the obtained results are reported and discussed. In Chapter 4, the conclusion of the thesis is made and future research topics are elucidated.

CHAPTER 2

DEEP NEURAL NETWORKS

Deep learning which is a subfield of machine learning, uses the power of multi-layered structure of algorithms called neural networks which are inspired by the functioning of a human brain to solve various problems in several fields including computer vision, speech recognition and natural language processing. With the help of increased computational power, the application of deep learning algorithms have become more prevalent.

2.1 Artificial Neural Networks

Neural networks, known as artificial neural networks (ANNs), are composed of nodes which are representative of biological neurons in animal brains. Each node produces a single output value which is the weighted sum of all inputs connected to that particular node. The output value is, then, transmitted to another node, ensuring the information flow. Besides the input and output layers, neural networks include hidden layers between them. In Figure 2.1, an artificial neural network containing two hidden layers, each having four nodes, is depicted.

In 1958, Rosenblatt [25] proposed *the Perceptron* which is considered as the earliest ANN. It acts as a binary linear classifier. It has inputs and each input, $x_i \in \mathbb{R}, i = 1, 2, \ldots, n$, has an associated connection weight, $w_i \in \mathbb{R}$. The first task in the perceptron is calculating the weighted sum of inputs:

$$y = \sum_{i=1}^{n} w_i x_i + w_0, \tag{2.1}$$

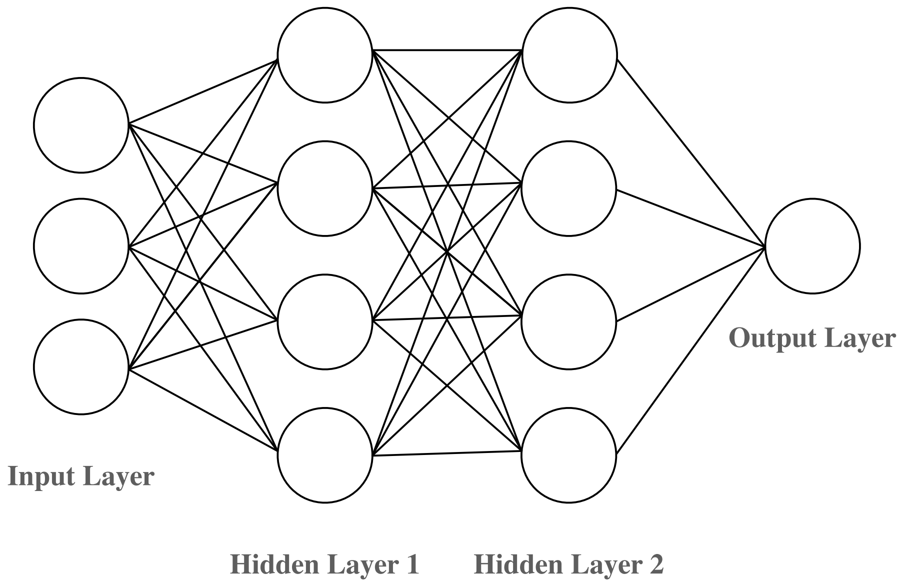


Figure 2.1: A diagram of artificial neural network with two hidden layers.

where y is the weighted sum and w_0 is the intercept value and considered as the associated weight of a bias unit, x_0, which is $+1$. The sum can also be written as a dot product:

$$y = \boldsymbol{w}^T\boldsymbol{x}, \tag{2.2}$$

where $\boldsymbol{w} = (w_0, w_1, \ldots, w_n)^T$ and $\boldsymbol{x} = (1, x_1, \ldots, x_n)^T$ are the vectors containing all of the weights and inputs including the bias weight and its corresponding input. Since the perceptron maps its input $\boldsymbol{x}$ to a single binary value, the next step is to use a threshold step function serving as an activation function:

$$\phi(a) = \begin{cases} 1, & \text{if } a > 0, \\ 0, & \text{otherwise,} \end{cases} \tag{2.3}$$

where ϕ represents a threshold step function and a is the input of the function. A schematic diagram describing the perceptron is shown in Figure 2.2.

Equation (2.1) defines a hyperplane that can be used to separate the input space into two regions. If the learning set is linearly separable, which means there exists at least one hyperplane that can classify the data that has two class labels, the weights

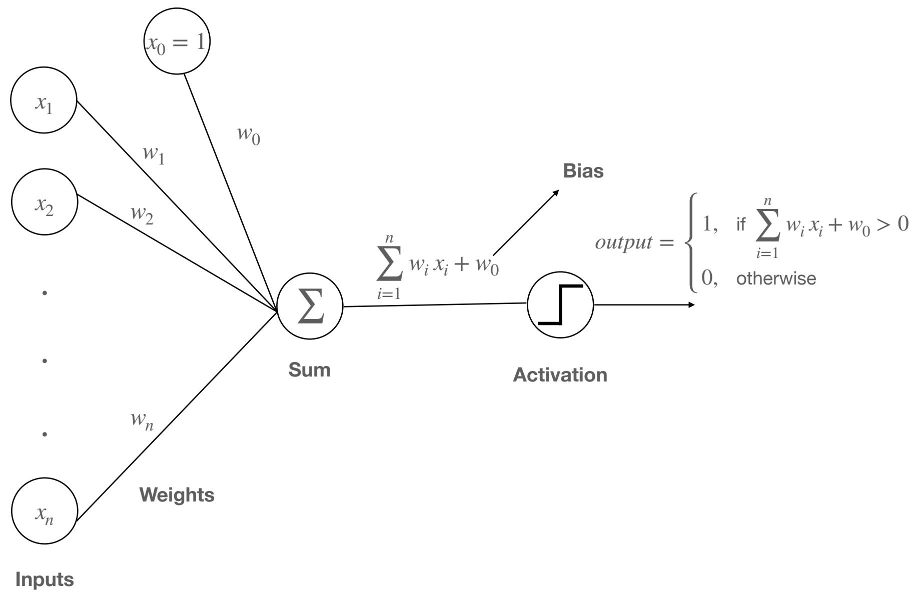

$x_0 = 1$
x_1
x_2
x_n
w_0
w_1
w_2
w_n
Bias
$\sum_{i=1}^{n} w_i x_i + w_0$
$output = \begin{cases} 1, & \text{if } \sum_{i=1}^{n} w_i x_i + w_0 > 0 \\ 0, & \text{otherwise} \end{cases}$
Sum
Activation
Weights
Inputs

Figure 2.2: A single layer perceptron.

and bias term can be learned such that the perceptron can accurately discriminate the training set.

2.1.1 Deep Feedforward Networks

Deep Feedforward Networks, or multilayer perceptrons (MLPs), are the deep learning models in which the information moves in only one direction, i.e., forward, from the input nodes to the output nodes. MLPs consist of an input layer, at least one hidden layer, and an output layer. Each node, except for the input layer nodes, uses a nonlinear activation function because such transformation of the input is needed to approximate a nonlinear function. It is mathematically proved that any feedforward neural network with at least one hidden layer has the ability to approximate any Borel measurable function [16]. In this regard, multilayer feedforward networks behave as universal approximators. These networks are generally used for supervised learning tasks where the purpose is to learn a function that maps an input to an output based on a set of training examples. Each example is a pair consisting of an input and corresponding output vector, the latter is usually called as the label. A feedforward neural network takes an initial signal, $\boldsymbol{x} \in \mathbb{R}^n$ and outputs $\boldsymbol{y} \in \mathbb{R}^m$, where n and m refer respectively to dimensions of input and output space.

Before introducing the computation of the forward pass of a feedforward neural network, there are some definitions need to be made. Let S_k represent the state of the kth layer

$$S_k = \sigma(W_k S_{k-1} + \boldsymbol{b}_k),$$

where W_k is the matrix corresponding to all the weights for layer k, $\boldsymbol{b}_k$ indicates the vector of biases for layer k. The matrix W_k can be expressed as:

$$W_k = \begin{bmatrix} w_{1,1} & w_{2,1} & \dots & w_{f,1} \\ w_{2,1} & w_{2,2} & \dots & w_{f,2} \\ \vdots & \vdots & \vdots & \vdots \\ w_{f,1} & w_{f,2} & \dots & w_{f,p} \end{bmatrix},$$

where, p and f are the number of nodes in layer k and $k-1$, respectively and $w_{i,j}$ is the weight connecting the ith node of layer $k-1$ to the jth node of layer k.

In order to find the next state, an activation function denoted as σ is applied. The above expression shows the dependency of the state of a particular layer on the previous one. Therefore, the calculation starts from the input layer and proceeds to the output layer:

$$\begin{aligned} S_1 &= \sigma(W_1 S_0 + \boldsymbol{b}_1) \\ S_2 &= \sigma(W_2 S_1 + \boldsymbol{b}_2) \\ S_3 &= \sigma(W_3 S_2 + \boldsymbol{b}_3) \\ &\vdots \\ S_L &= \sigma(W_L S_{L-1} + \boldsymbol{b}_L) \end{aligned}$$

Here, S_0 is the input vector $\boldsymbol{x}$ and S_L is the output vector $\boldsymbol{y}$, where L denotes the number of layers (excluding input layer). Once the forward pass is completed, the error between the ground truth (target values) and the output of the network is calculated. Next step is the learning process where the weights and biases are updated such that the error would be decreasing.

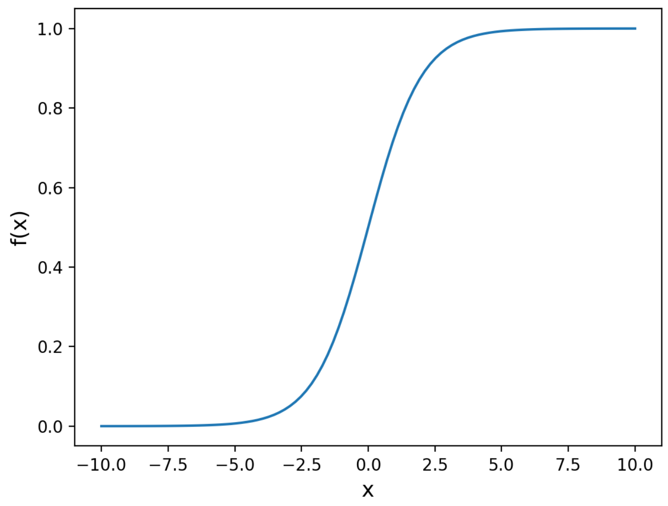


Figure 2.3: Sigmoid (Logistic) function.

2.1.2 Activation Functions

In ANNs, the activation function, also known as the transfer function, is applied to a neuron to calculate its output response. There are linear activation functions as well as nonlinear activation functions. Nonlinear ones are used more often to increase the learning power of a neural network when it comes to finding the complex relationship between the input and output data.

2.1.2.1 Sigmoid Function

The sigmoid function, or the logistic function, is a function that is represented by an "S"-shaped curve and is defined by the formula:

$$f(x) = \frac{1}{1 + e^{-x}}. \tag{2.4}$$

The sigmoid function, which is differentiable and bounded, maps any real value to the range of 0 and 1. The graphical representation of the sigmoid function is shown in Figure 2.3.

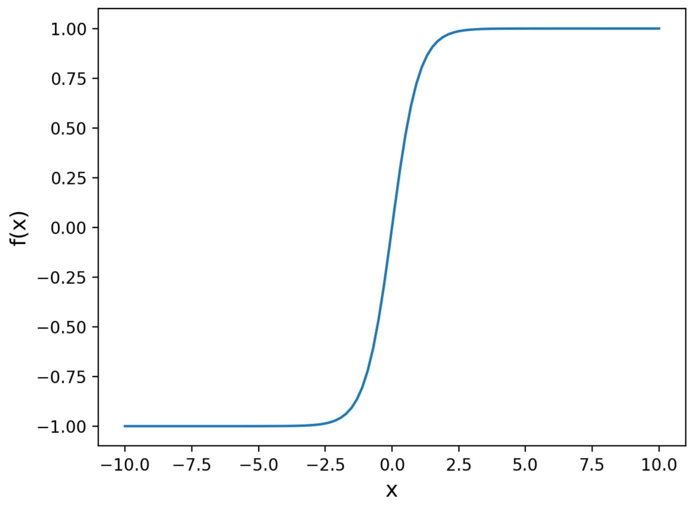


Figure 2.4: Hyperbolic Tangent (tanh) function.

2.1.2.2 Hyperbolic Tangent Function

The hyperbolic tangent activation function, or tanh for short, has also an "S"-shaped curve like the sigmoid function. Contrary to the sigmoid function, the range of the tanh function is from -1 to 1. The tanh function is given as:

$$f(x) = \tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{2.5}$$

The tanh function produces zero-centered outputs because its range contains both negative and positive values. This property provides an advantage when optimizing a neural network's weights compared to using an activation function like the sigmoid that has not not zero-centered outputs. Figure 2.4 shows the tanh function.

2.1.2.3 Rectified Linear Unit (ReLU) Function

The ReLU, or the rectifier, is an activation function commonly used in deep neural networks, especially in convolutional neural networks. If the input is positive the output will be equal to the input, otherwise, the output will be 0. Thus, the function can be written as:

$$f(x) = \max\{0, x\}. \tag{2.6}$$

As it can be seen in Figure 2.5, the ReLU is a piecewise linear function and not

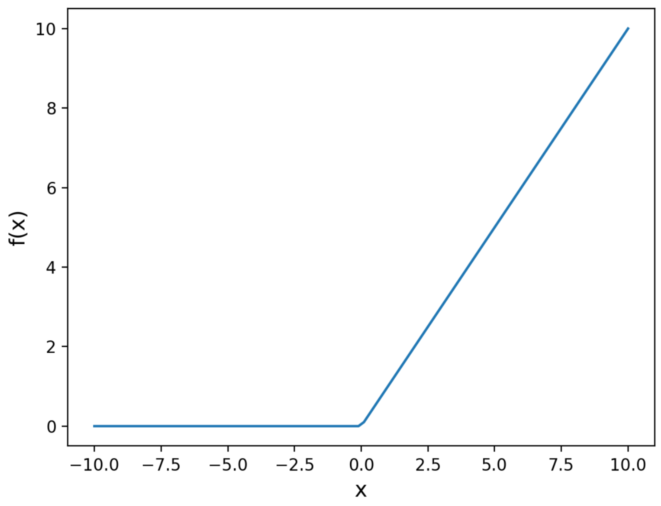


Figure 2.5: Rectified Linear Unit (ReLU) function.

differentiable at 0.

The ReLU activation function does not suffer from the vanishing gradient problem compared to the sigmoid and tanh functions [17]. The problem arises when calculating derivatives of the activation functions that have high input values. Contrary to sigmoidal functions whose derivatives approach 0 as the input value gets higher, ReLU has a constant derivative of 1. This makes ReLU a proper choice for many deep learning applications.

2.1.3 Optimization of Deep Neural Networks

As described earlier, a deep neural network acts as a composite function from input to output space. The parameters of the function, also called as weights, controls the function, i.e., the network, behavior regarding how good the fitting is for a given training data. The learning process is finding the best combination of weights such that the error function (also called as the loss function) will be minimized. For that, one needs to apply an optimization algorithm.

In machine learning, gradient descent based algorithms are commonly used to train the models. The principle of gradient descent is to update the weights and biases by moving repeatedly in the opposite direction of the gradient, i.e., the derivative of the loss function with respect to the parameters we wish to update. Let $J(\boldsymbol{\theta}) : \mathbb{R}^n \to \mathbb{R}$ be a cost function and $\boldsymbol{\theta}$ be the parameters to be updated. The updating mechanism according to gradient descent algorithm is as follows:

$$\boldsymbol{\theta}_{new} = \boldsymbol{\theta}_{old} - \eta \nabla J(\boldsymbol{\theta}_{old}), \tag{2.7}$$

where η is the learning rate, or the step size of the algorithm. The learning rate is a positive scalar that determines how big a step is taken in the descent direction. However, care should be taken in setting the rate. If it is too high, there will be a risk of overshooting the minima. In the opposite case, the rate of convergence will be low or the algorithm might get stuck in an unwanted local minimum [6]. In (2.7), $\nabla J(\boldsymbol{\theta}_{old})$ represents the gradient vector whose components are the partial derivatives of J at $\boldsymbol{\theta}_{old}$.

For updating the weights of a network, the derivative of the error term with respect to each weight should be calculated. One effective algorithm for this task is backpropagation which employs the chain rule while performing a backward pass to adjust the model's weights and biases [26].

If the entire training data is used to compute the gradient of the loss function for the parameter updating, the algorithm is called batch gradient descent (BGD). In this type of optimization, each iteration is referred to as an epoch as the all training examples are taken into account for a single model update. For a huge amount of data, the computation time of the BGD algorithm can be very long.

Another variant of gradient descent, namely stochastic gradient descent (SGD) in which only one training example is used for each update, can be implemented to overcome this problem. Stochasticity reveals itself when calculating the derivative. Instead of the actual graident calculated from the entire dataset, in stochastic variants, an approximation of the true gradient calculated from a randomly selected training sample or a subset of the whole dataset is used. Let N be the number of total training examples. The parameter updating in SGD is given as:

$$\boldsymbol{\theta}_{new} = \boldsymbol{\theta}_{old} - \eta \nabla J(\boldsymbol{\theta}_{old}; x^{(i)}; y^{(i)}), \tag{2.8}$$

where $x^{(i)}$ and $y^{(i)}$ represents the ith training input and the corresponding label, respectively. The update expressed by (2.8) is repeated for every i from 1 to N.

However, the most commonly applied approach is splitting the data according to a predetermined number, called as batch size, into small batches of examples and performing an update for every batches. This method is known as mini-batch gradient descent (MBD) and enables us to benefit from the strengths of both BGD and SGD. Since it uses more data points to approximate true gradients compared to SGD, updating steps are less noisy, which decreases the number of iterations for an acceptable convergence. Moreover, MBD surpasses BGD in terms of the training speed, especially for large datasets.

2.1.4 Adam Optimization

Adam [19] (short for Adaptive Moment Estimation) is an optimization algorithm and can be considered as an extension to SGD.

Algorithm 1: Adam Optimization Algorithm

```
Input: Exponential decay rates for moments estimate $\beta_1$, $\beta_2$
Input: Learning rate (stepsize) $\alpha$
Input: Loss function with parameters $\theta$ $\mathcal{L}(\theta)$
$\theta_0 \leftarrow 0$ (Initialize parameters)
$m_0 \leftarrow 0$ (Initialize first moment)
$v_0 \leftarrow 0$ (Initialize second moment)
$i \leftarrow 0$ (Initialize timestep)
while $\theta_i$ not converged do
    $i \leftarrow i + 1$
    $g_i \leftarrow \nabla\mathcal{L}_i(\theta_{i-1})$ (Calculation of gradients at timestep $i$)
    $m_i \leftarrow \beta_1 \cdot m_{i-1} + (1 - \beta_1) \cdot g_i$ (First moment update)
    $v_i \leftarrow \beta_2 \cdot v_{i-1} + (1 - \beta_2) \cdot {g_i}^2$ (Second moment update)
    $\hat{m}_i \leftarrow m_i/(1 - {\beta_1}^i)$ (Computation of bias-corrected first moment)
    $\hat{v}_i \leftarrow v_i/(1 - {\beta_2}^i)$ (Computation of bias-corrected second moment)
    $\theta_i \leftarrow \theta_{i-1} - \alpha \cdot \hat{m}_i/(\sqrt{\hat{v}_i} + \epsilon)$ (Parameters update)
end
```

In this algorithm, an adaptive learning rate is calculated for each parameter update, meaning that the need for choosing a learning rate manually for each learning session is eliminated. The algorithm for Adam optimization is given in Algortihm 1.

2.2 Convolutional Neural Networks

Convolutional neural networks (CNNs), developed by LeCun et al. [20] in the 1980s, are a specialized class of deep neural networks. They are commonly used in processing data that has a grid-like topology, such as images [13]. As the name "convolutional" indicates, these types of neural networks use a linear mathematical operation called *convolution* which is extensively employed in signal and image processing. In mathematics, convolution can be considered as an operation on two functions such as:

$$h(t) = (f * g)(t) = \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau. \tag{2.9}$$

Here, f and g are two real-valued functions of τ and t is a fixed parameter. The symbol $*$ denotes the convolution operation that is defined as the integral of the product of the two functions, namely $f(\tau)$ and $g(t-\tau)$ which is a reversed version of $g(\tau)$ that has been shifted by an amount t. The resulting output function is represented as $h(t)$. In the realm of machine learning, the function f and g are often called as the *input* and *kernel*, respectively. The function h, on the other hand, is referred to as the *feature map*.

CNN based applications usually use multidimensional arrays of data. For example, a grayscale image can be considered as a two-dimensional array of data consisting of integer values, each representing the intensity value of the corresponding pixel. Thus, writing the discrete form of the convolution operation would be more appropriate for this case. For a given two-dimensional input image I and a two-dimensional kernel K, (2.9) can be written as:

$$F(i,j) = (I * K)(i,j) = \sum_a \sum_b I(a,b)K(i-a, j-b). \tag{2.10}$$

Here, F represents the output feature map. The commutative property of the convolution operation allows us to write (2.10) in the form of:

$$F(i,j) = (K * I)(i,j) = \sum_a \sum_b I(i-a, j-b)K(a,b). \tag{2.11}$$

The indices i and j are related to the input image I, whereas the indices a and b are related to the kernel K. In many machine learning applications, an operation called *cross-correlation* that does not require flipping the kernel, as opposed to convolution, is used:

$$F(i,j) = (I * K)(i,j) = \sum_a \sum_b I(i+a, j+b)K(a,b). \tag{2.12}$$

Although cross-correlation differs from convolution in a mathematical sense as described above, it is conventionally called convolution in CNN based applications. A visual example of the convolution operation applied to a two-dimensional image is given in Figure 2.6. The calculation of the first element in the feature map shown in Figure 2.6 by using (2.12) can be explicitly stated as:

$$\begin{aligned}
F(0,0) &= \sum_{a=-1}^{a=1} \sum_{b=-1}^{b=1} I(0+a, 0+b)K(a,b) \\
&= I(-1,-1)\cdot K(-1,-1) + I(-1,0)\cdot K(-1,0) + I(-1,1)\cdot K(-1,1) \\
&\quad + I(0,-1)\cdot K(0,-1) + I(0,0)\cdot K(0,0) + I(0,1)\cdot K(0,1) \\
&\quad + I(1,-1)\cdot K(1,-1) + I(1,0)\cdot K(1,0) + I(1,1)\cdot K(1,1) \\
&= 9\cdot(-4) + 1\cdot 0 + 3\cdot 1 \\
&\quad + 8\cdot 2 + 7\cdot(-1) + 4\cdot(-3) \\
&\quad + 6\cdot 7 + 4\cdot 4 + 2\cdot 2 \\
&= 26,
\end{aligned}$$

where $K(0,0)$ is located at the center of the kernel. Similarly, $I(0,0)$ is the center element of the overlapping portion of the input with the kernel. Since the size of the kernel is 3×3, a and b take values -1, 0, and 1.

CNNs differ from MLPs in terms of the connectivity structure of the neurons. MLPs are composed of fully connected layers, that is, every single neuron in a layer is connected to all the neurons in the previous layer. Contrary to MLPs, CNNs have convolutional layers that are sparsely connected because the convolutional kernel size is smaller than the input.

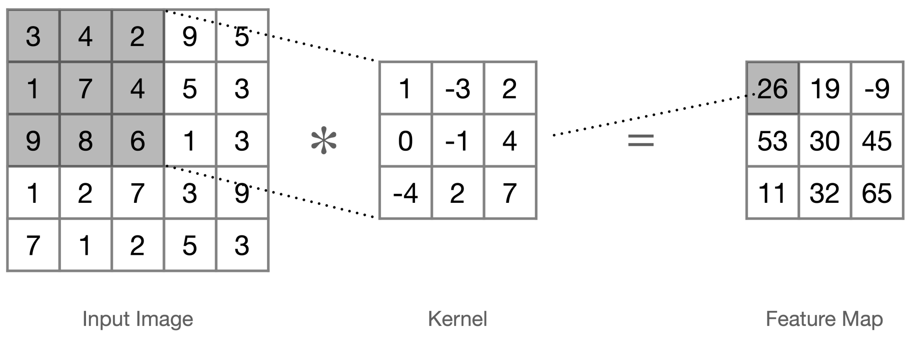


Figure 2.6: An illustration of a convolution operation applied to a given two-dimensional input image. The 3×3 kernel shifts over the 5×5 input image, producing each pixel of the feature map.

2.2.1 Convolutional Layers

In a CNN, convolutional layers are the main building blocks of the model architecture. The layer takes the input data which is considered as a tensor that has height, width, and channel dimensions. Then, a filter, also known as kernel, moves over the input while carrying out the dot product operation with the overlapping patch of the input. The operation is done such that the matching elements are multiplied, then these multiplications are summed up. The resulting scalar values create the feature map that will be the input of the next layer. A nonlinear activation function, such as ReLU activation, is applied to each entry in the feature map, just like in the case of a fully connected layer.

The convolutional filter must have the same number of channels as the input. The filter entries are the parameters that are to be learned by means of an optimization method. However, single filter is not sufficient enough to capture multiple features of the input. Hence, many filters are simultaneously used to extract different features from the input and the channel dimension of the output is equal to the number of filters used.

As it can be seen in Figure 2.6, when the convolution operation is performed on an image, it reduces the size of the image. Therefore, the image gets shrunk as the number of convolutional layers applied to it increases, preventing us to build deeper networks. Besides, corners and edges interact less with the kernel, so the information coming

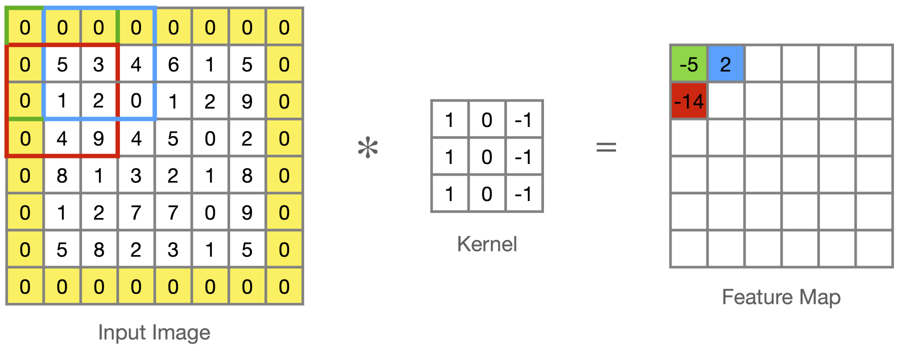


Figure 2.7: A schematic representation of convolution of a 6×6 input image. In order to preserve the spatial size of the input, zero-padding is applied by adding an additional layer of zeros around the borders, making the size of the padded image equal to 8×8. A filter of size 3×3 is used for the convolution operation and it moves one pixel at a time over the input, i.e., the stride number is 1. The resulting feature map has the same size as the input image.

from these regions tends to get lost. To alleviate the stated problems, a concept called padding is used. Padding is achieved by adding zeros around the borders of the input, enabling us to control the output volume. In most cases, it is used to conserve the width and height of the input. The simple formula for calculating the height/width dimensions of the output of a convolutional layer is given as:

$$O = \frac{W - K + 2P}{S} + 1, \tag{2.13}$$

where O, W, and K indicate the height/width dimensions of the output, input, and filter, respectively. P is the padding amount and S denotes the stride number. Stride is the amount by which the filter shifts over the input image.

Think of an input image of size 16×16×3, and 8 filters of size 3×3×3 which are to be applied to convolve the image. The output channel is equal to the number of filters and, in this case, it is 8. With single stride and no padding, the output dimensions will be 14×14×8. If, however, zero-padding is applied to the input image in such a way that an additional layer of zeros is added around the borders of the image, the output dimensions will be 16×16×8 preserving the spatial dimensions of the input. In Figure 2.7, it is shown that the application of zero-padding with a proper stride size would result in an output having the same dimensions as the input.

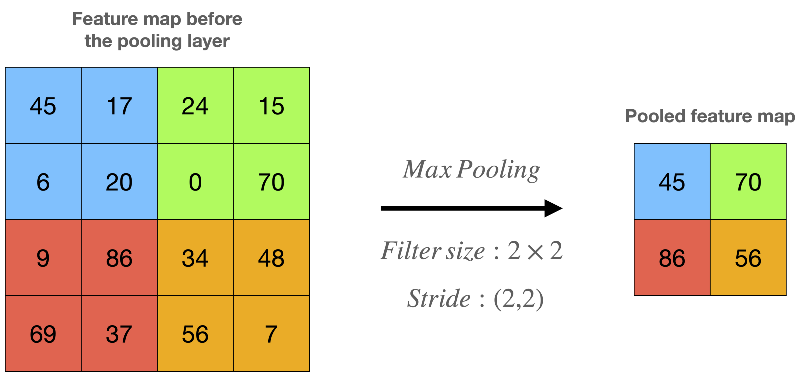


Figure 2.8: Example of max pooling operation.

2.2.2 Pooling Layers

Another type of layer that is used immediately after a convolutional layer is a kind of subsampling layer and called as the pooling layer. The main function of pooling layers is to reduce the dimensions of the feature maps, resulting in a decrease in the number of learnable parameters and the computational cost. The pooling operation is applied for each feature map, preserving the channel dimension of the input. There are two commonly used types of pooling operations:

Max Pooling: This type of pooling operation works by selecting the maximum element of a patch covered by the pooling filter in the feature map. The selected elements are used to generate the downsampled feature map. An example of the max pooling operation is illustrated in Figure 2.8.

Average Pooling: Average pooling operation is done by calculating the average of the elements in a patch covered by the pooling filter. Hence, the pooled feature map consists of the average features of each patch presented in the feature map before the pooling operation is applied. Figure 2.9 visualizes an application of average pooling operation to a 4×4 feature map.

In this chapter, we have explained the concept of ANNs by firstly introducing the perceptron which is a single-layer neural network, that constitutes the building block of multilayer feedforward neural networks. Then, CNNs which can be considered as a special class of feedforward neural networks have been discussed to give a detailed insight into the theory behind the work of this thesis.

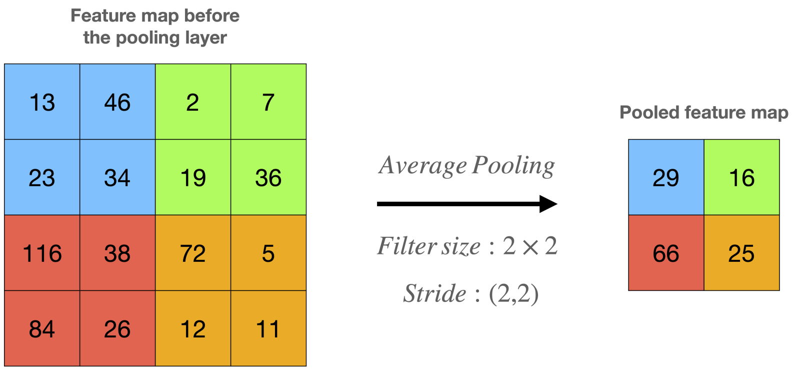


Figure 2.9: Example of average pooling operation.

CHAPTER 3

DENOISING OF MICRO-COMPUTED TOMOGRAPHY IMAGES OF ROCK SAMPLES

CNNs are capable of extracting spatial features from the data, which makes them suitable for image-related applications. That is why CNN-based methods have been extensively and successfully used in various image processing tasks including image denoising.

In this work, we use the power of CNNs to learn a nonlinear mapping from a reduced exposure time micro-CT image of rock sample to its corresponding high exposure time image. The method is completely data-based and does not require prior knowledge about the noise statistics.

As a CNN model, we use U-net architecture, which was originally developed for biomedical image segmentation. The architecture is explained in detail in this chapter along with the datasets used in this work. The methodology of the application and the obtained results are discussed to evaluate the efficacy of the method.

3.1 Preparation of the Datasets

In this work, real micro-CT datasets generated by using FEI HeliScan MicroCT imaging system were used[1]. Since the aim is to improve the quality of low-dose scans which are obtained by reducing exposure time, two datasets were acquired; one of them consists of scans produced at an exposure time of 0.5 s per projection, and for

[1] Dataset is provided by Dr. Guenther Glatz, King Fahd University of Petroleum and Minerals.

the other one the exposure time was taken as 1.4 s per projection. The scanning was configured such that there were 1800 projections per revolution and the X-ray tube voltage and current were of 85 kV and 72 mA, respectively. The total acquisition time per revolution was 42 min for the high exposure time experiment, whereas reduction of the exposure time caused more than 60 % decline and resulted in 15 min for the low exposure time experiment.

A longer exposure time leads to less noisy images because more photons are collected at the detector. Therefore, the scans collected at 1.4 s constitute the dataset of high quality images. The other dataset is composed of low quality images procured by setting the exposure time as 0.5 s. A porous carbonate rock specimen having a diameter of 1.5 inches and a length of about 2 inches was used for the scanning. The resolution of each obtained image is 2884×2884 pixels and its data type is unsigned 16-bit meaning every pixel in it can take values ranging from 0 to 65535. It is inefficient to use a 2884×2884 image for training a neural network. For this reason, by extracting two patches of size 512×512 pixels at locations around the middle of each image, a total of 2000 images were prepared for each dataset.

Preprocessing of the raw data is a crucial step before training the model. It involves all the processes that transform the raw data and data normalization is one of them. In computer vision tasks, where images are used, data normalization refers to scaling the pixel values so that we can obtain faster learning process. There are several methods including min-max normalization, zero-mean normalization and decimal scaling for data normalization.

In our case, min-max normalization was applied to the data before using it in training the network. For this type of normalization, a simple formula is used:

$$\frac{value - min}{max - min}. \tag{3.1}$$

Normalization was done for both datasets separately, as a result, the minimum pixel value in a particular dataset got transformed into 0, and the maximum pixel value into 1. Figure 3.1 shows a random pair from the prepared datasets. The pair consists of a low exposure time (0.5 s) image and the corresponding high exposure time (1.5 s) image. Higher noise level can be visually detected in the low quality image.

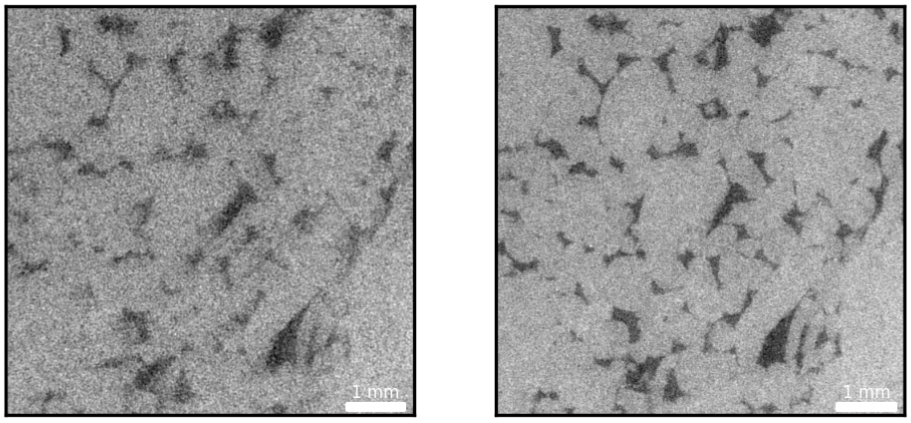


Figure 3.1: A random pair of images representing both datasets. The left one is from the low exposure time (0.5 s), low quality, data and the right one is from the high exposure time (1.5 s), high quality, data.

3.2 Network Architecture

A U-Net [24] based deep convolutional neural network was used to achieve quality enhancement in micro-CT images by training the network in a supervised learning framework. The architecture of the model contains two paths connected by a bridge section.

First path is the contraction path which is used to capture the context in the image. This path is also called as the encoder. The encoder consists of three blocks of convolution layers. In each block, three consecutive convolution layers are applied followed by ReLU activation function after each layer. At the end of each block, max pooling operation is applied to reduce the size of the feature map.

The second path is the expanding path (also called as the decoder). Transposed convolution layer is used in each block before three consecutive convolution layers. The parameters for each transpose convolution are arranged such that, the size of the image are doubled while the number of channels (depth) is halved. At every step of the decoder, skip connections were added by concatenating the output of the transposed convolution layers with the feature maps from the encoder at the same level.

The network is an end-to-end fully convolutional network (FCN), that is, it only contains convolutional layers and does not contain any dense layers as in typical CNNs.

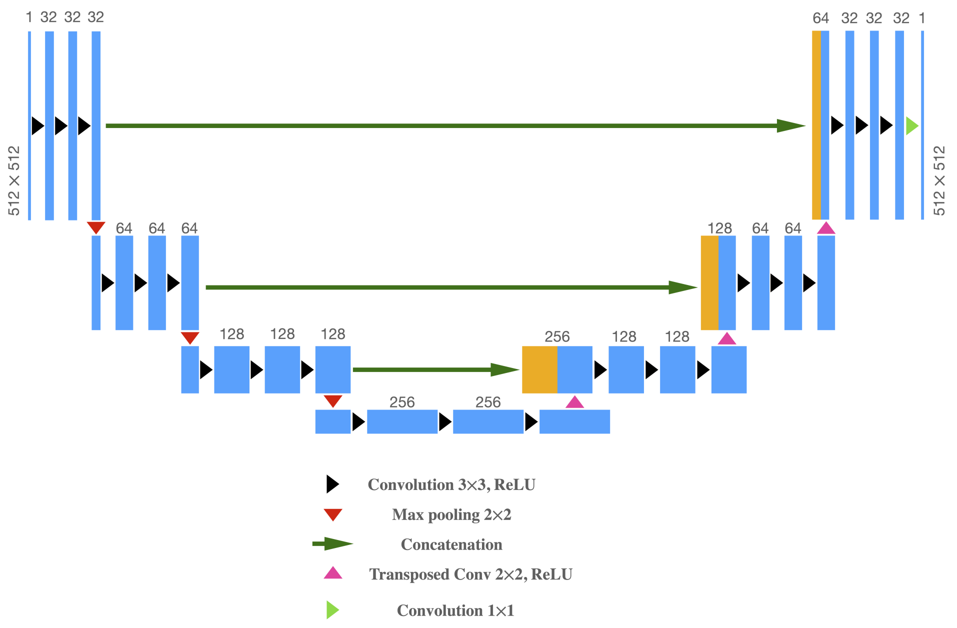


Figure 3.2: Schematic representation of the proposed DCNN composed of two main paths: encoder path and decoder path. Each blue box is a feature map with number of channels attached to its top. The input image is of size 512×512. The network architecture was designed to make the dimension of output image same as input one. Skip connections are denoted by long green arrows.

The kernel size used in each convolution layer is 3×3 and the number of filters starts from 32 and increases up to 128 in the encoder stage. On the contrary, in the decoder stage the number of filters used in the convolution layers starts from 128 and decreases to 32 preserving the symmetry in the network architecture. Figure 3.2 depicts the whole framework of the network used in this work.

3.3 Loss Functions

A loss function measuring the discrepancy between the network's predictions and the target values, the high exposure time scans in this case, needs to be defined so that it should be minimized during the learning. Two different loss functions were used, namely, MSE and SSIM to see the impact of them on the performance of the network.

MSE is one of the most prevalent loss functions used for regression problems. In computer vision tasks, it is computed by averaging the squared intensity differences of pixels of the images that are to be compared. The MSE loss function for a patch

P, having N pixels, can be written as in [32]:

$$\mathcal{L}^{MSE}(P) = \frac{1}{N} \sum_{p \in P} (h(p) - n(p))^2, \tag{3.2}$$

where p represents a pixel value in the patch P, $h(p)$ and $n(p)$ are the values of pixels in the high quality reference patch and the network's prediction patch, respectively.

SSIM, proposed by Wang et al. [28], is an image quality metric and used for measuring the similarity between two images. Consider that two aligned patches extracted from two images which we want to measure the similarity between them. Let x and y denote these two patches. The SSIM index is given as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x \mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{3.3}$$

where μ_x and σ_x^2 are the mean and variance of x, respectively. Similarly, μ_y and σ_y^2 refer to the mean and variance of y, respectively. The covariance of x and y is expressed by σ_{xy}. To avoid instability when terms in the denominator are close to zero, the constants C_1 and C_2 are added to the expression.

To evaluate the overall image quality, one needs to calculate the SSIM index for each window moving across the whole image and find the average of them. In other words, a mean SSIM (MSSIM) index is used to determine the similarity between two images. For example, let X and Y represent the reference and the distorted images, respectively. The following expression gives how close Y is to X in terms of structural similarity:

$$\text{MSSIM}(X, Y) = \frac{1}{M} \sum_{j=1}^{M} \text{SSIM}(x_j, y_j). \tag{3.4}$$

Here, M is the number of windows. For each local window the SSIM index is calculated using the image contents x_j and y_j at the jth window. Therefore, the SSIM loss function for a patch P can be written as

$$\mathcal{L}^{\text{SSIM}}(P) = 1 - \text{MSSIM}(h, n). \tag{3.5}$$

SSIM is a bounded metric with the maximum possible value of 1. For the two identical images, the resulting SSIM index is 1, so the similarity is expected to be higher as the index value gets closer to 1. That is why the index value is subtracted from 1 in the loss term.

3.4 Network Training

In this section, training of the network will be explained in details. Before training the network, the data was separated into training, validation, and test sets. In supervised learning tasks, the aim is to learn a mapping from an input to an output (label), so there needs to be a set of labeled training examples (input-output pairs) to train the network. Initially, there were 2000 low quality images which can be considered as the input data and the corresponding high quality ones, i.e., output data. Firstly, 80% of the available data was allocated for training. Subsequently, a further split was applied on the training set to reserve 20% of it as a validation set. This is a common data splitting approach adopted for training and evaluating a deep learning model.

Besides, in order to see the effect of selecting different validation and training datasets on the performance of the model, k-fold cross-validation method was used. In k-fold cross-validation, the dataset is partitioned into k equal folds. One of the folds is kept for validation purposes and the remaining $k - 1$ folds are used for training. Then, the model is trained for k times, giving each fold a chance to be the validation set. We applied 5-fold cross-validation technique to evaluate the model's performance on the 5 different validation sets. Each fold consists of 320 images, which means that 1280 images are used for training and 320 images are used for validation for each model training. The results are presented in Table 3.1.

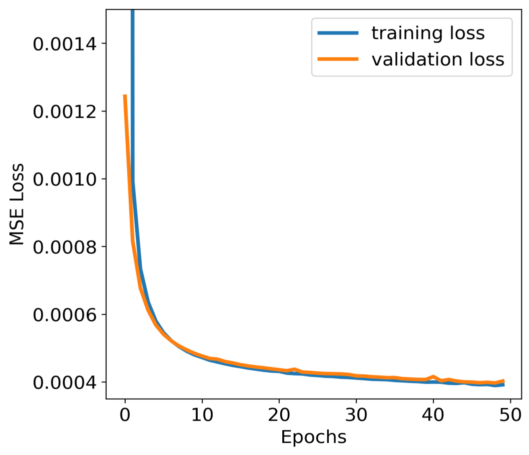
In both the encoder and the decoder paths of the network architecture, each block involves three consecutive convolution layers. Actually, we can think of the number of convolution layers in each block as one of the hyperparameters that can control the model's learning ability. Before deciding to put three layers, two layers scenario was also implemented. However, for our problem, the obtained qualitative results convinced us to use three convolution layers in each block.

Before starting the training, the weights, that are to be updated during the optimization, should be initialized. For layers and nodes using the ReLU activation function, generally He [15] initialization method is used. Therefore, the convolutional kernel weights were initialized using this approach. It was observed that using different mini-batch sizes has no impact on the performance of the network for our problem. The mini-batch size was set to be 32, which means that the model weights are updated after 32 samples are fed into the network. Adam optimizer with a learning rate of 0.0001 was selected for the optimization of the network's parameters. The chosen learning rate is robust and does not require altering it while training because the adaptive nature of Adam algorithm assigns a distinct learning rate for each parameter updating. The hyperparameters were selected such that the model is not overfitting the training data, i.e., the validation loss does not increase at some point while the training loss continues to decrease. The model was trained on a tensor processing unit (TPU) provided by the Google Colab environment, a cloud-based Jupyter notebook platform. For the implementation of the model, TensorFlow's Keras module was used.

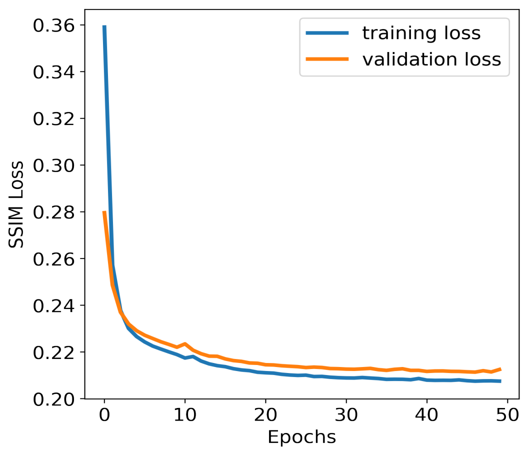
3.5 Denoising Results

The proposed network was trained with 1280 training examples to make it learn a mapping from an image with reduced exposure to its corresponding high quality one. A set of 320 pairs of images was set to be as validation to monitor the changes in the validation loss along with the training loss. The validation loss is not used for the gradient calculations to update the weights. It may be used to detect overfitting, which manifests itself when the validation loss starts to increase at some point while the training loss continues to fall. This indicates that the model is overfitting the training data and fails to generalize well to additional data, thus, damaging the predictive power of it. The learning curves of the model both for MSE and SSIM loss functions are depicted in Figure 3.3. The validation loss in both cases, decreases along with the training loss and does not start to increase after a certain number of epochs, which indicates that the model is not overfitting, therefore, is ready to be evaluated against the test dataset.

In Table 3.1, the evaluation results for 5-fold cross-validation method is given. The method was applied for both MSE and SSIM optimized networks. The validation



(a)



(b)

Figure 3.3: The generated learning curves during the training of the network using (a) MSE as loss function and (b) SSIM as loss function. Both plots include training and validation error curves.

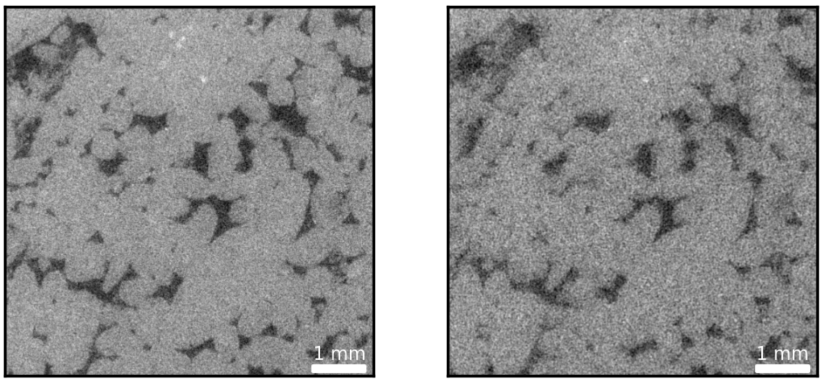
Table 3.1: 5-fold Cross-validation Results

Training set	**Validation set**	**Validation loss (MSE)**	**Validation loss (SSIM)**
Folds 2,3,4,5	Fold 1	3.80×10^{-4}	0.21
Folds 1,3,4,5	Fold 2	3.95×10^{-4}	0.21
Folds 1,2,4,5	Fold 3	3.83×10^{-4}	0.20
Folds 1,2,3,5	Fold 4	3.89×10^{-4}	0.20
Folds 1,2,3,4	Fold 5	3.82×10^{-4}	0.21

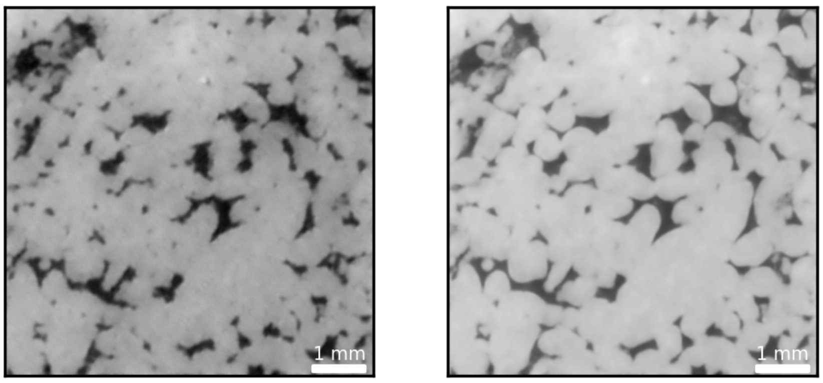
loss of each training is stated. As it can be clearly seen, the variance of the validation losses is too small. This indicates that changing the training set has no impact on the performance of the model, meaning that the hold-out method is a good choice in our case.

To assess the performance of the trained model on the unseen data, we use the test set which is composed of 400 example pairs. A randomly selected pair from the test set and the denoising outputs of the network based on both loss functions are shown in Figure 3.4. It is visually clear that the network is able to remove noise significantly regardless of the loss function being used while training.

Nevertheless, it was observed that while eliminating noise, different losses tend to highlight different features of the porous media. The network optimized by MSE loss was revealed to be more sensitive to fine scale pore space than the network optimized by SSIM loss. As it can be seen in Figure 3.4b, SSIM based reconstruction gives



(a)



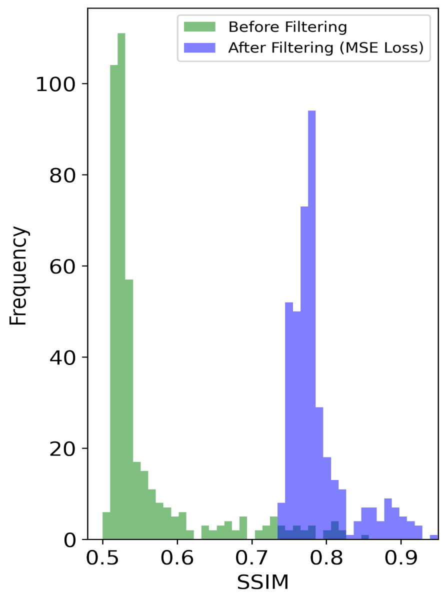
(b)

Figure 3.4: Denoising results of a randomly selected example from the test dataset. (a) A pair of images from the test set. The left one is a high exposure time scan (label) and the right one is the corresponding low exposure time scan (input). (b) The network denoising results. The left one is the reconstructed image predicted by the MSE optimized network and the right one is the result of the SSIM optimized network.

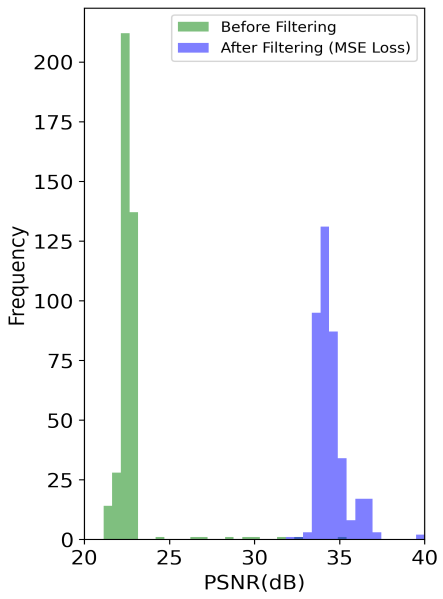
smoother textures and sharper boundaries.

For the quantitative assessment of the improvement, we use two different image metrics, namely, SSIM and PSNR. PSNR is a commonly used quality metric in image processing. It is defined as the peak signal-to-noise ratio between two images and expressed in decibels. It gives us a quantitative measure of how close a distorted image to the original one. The greater the value of PSNR, the better the quality of the distorted image is. The calculation of the PSNR is given by

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{\mathrm{MAX}_I^2}{\mathrm{MSE}}\right), \tag{3.6}$$



(a)



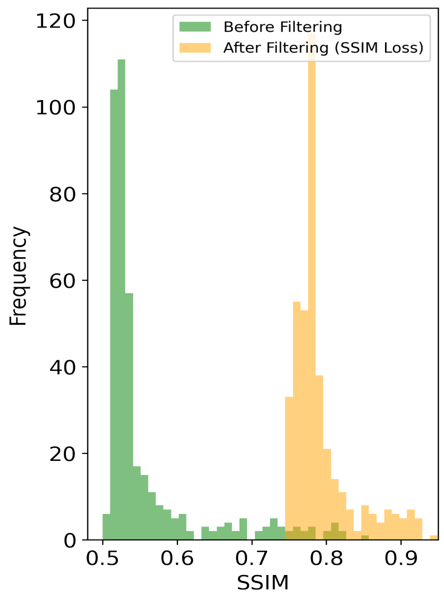
(b)

Figure 3.5: Histograms of the quality metrics calculated for the test set. The filtered images are the predictions of the network optimized by MSE loss. (a) Histogram of the SSIM values. (b) Histogram of the PSNR values.

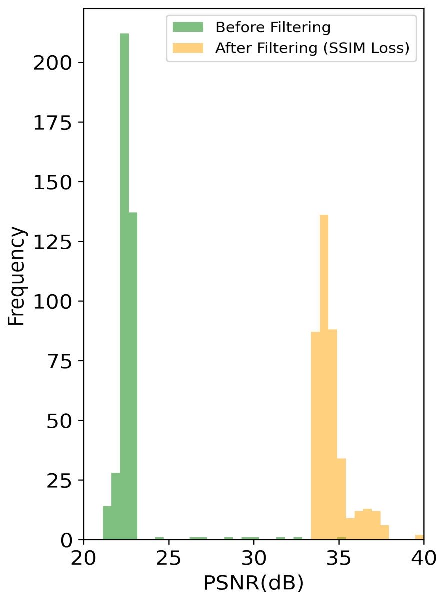
where MAX_I is the maximum possible pixel value.

In Figure 3.5 and Figure 3.6, histograms of both SSIM and PSNR values calculated for the test set are demonstrated. Each histogram shows the frequency distribution of the quality metric values computed with respect to the high exposure scans. The first distribution (before filtering) is indicating the values calculated for the low exposure scans and the second one (after filtering) shows the distribution of the values calculated for the denoised scans filtered by the trained network. The test dataset consisting of 400 image pairs was used for the evaluation and resulting metric values have shown remarkable success of the trained network. From a quantitative point of view, as can be seen in the histogram plots, both losses resulted in similar metric improvements. It is clear that the reconstructed images are far superior in quality to those with reduced exposure, as evidenced by the substantial increase in metrics calculated with respect to the high exposure images.

Furthermore, intensity profiles were taken along a horizontal line, which consists of a single row of 512 pixels, both on a random test pair and the corresponding reconstructed images to show the network's denoising power. Figure 3.7 shows the



(a)

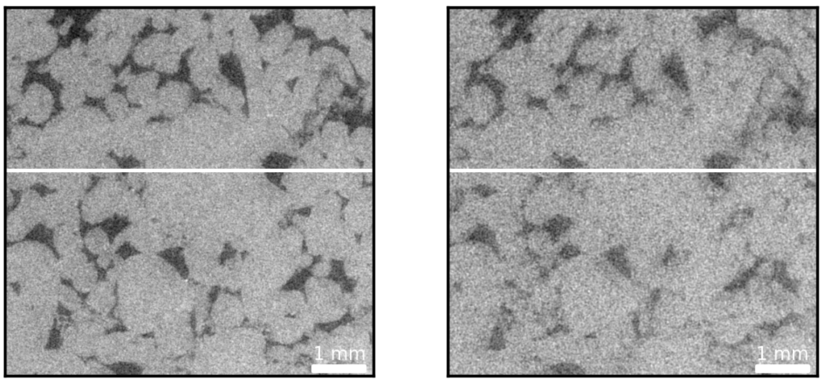


(b)

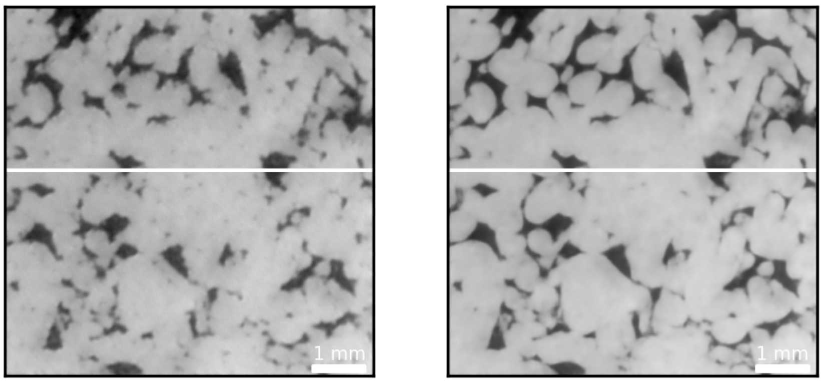
Figure 3.6: Histograms of the quality metrics calculated for the test set. The filtered images are the predictions of the network optimized by SSIM loss. (a) Histogram of the SSIM values. (b) Histogram of the PSNR values.

random pair from the test set and the corresponding outputs suggested by both MSE and SSIM optimized network. The white lines are the cross sections along which the profiles are taken. The pixel intensity profiles are depicted in Figure 3.8 in order to show the success of the trained network to denoise a given low exposure time scan. More importantly, the network is able to distinguish noise from the other high frequency contents like edges, corners, and sharp structures, thus, retaining the critical components of the image.

Surprisingly, if we look at the scans predicted by the network, we see that they are not only superior than the low-dose scans (inputs) but also greater quality compared to the high exposure time scans (labels). Since, because of the nature of a CT scanning, there is some level of noise associated even with the images collected at an exposure time of 1.4 s (labels), we do not have an absolute reference, i.e., ground truth, to verify the improvement. Therefore, we further create an artificial dataset based on the images denoised by the network to validate the results obtained by training the proposed architecture. We present this in the next section.

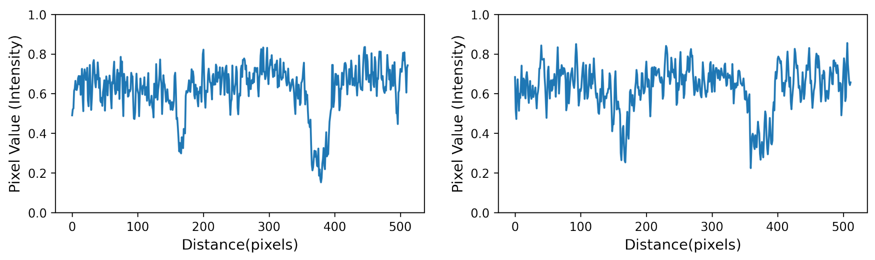


(a)

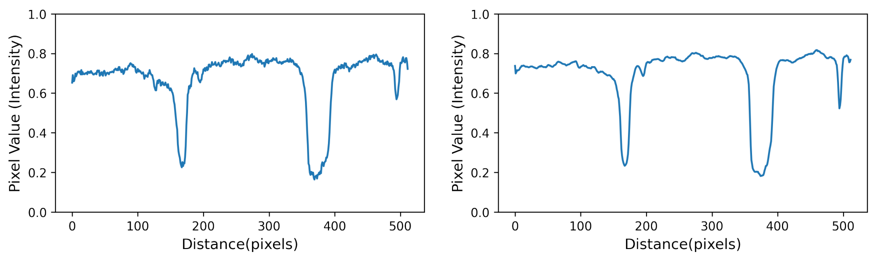


(b)

Figure 3.7: A random pair from the test set and the denoising results predicted by the trained network. (a) A high exposure time scan (left) and the related low exposure time scan (right). (b) Predicted images by the network trained using the MSE loss function (left) and the SSIM loss function (right). The cross-sectional profiles were taken along the white horizontal lines depicted on each image.



(a)



(b)

Figure 3.8: Pixel intensity profiles taken along the lines indicated in Figure 3.7. (a) The profiles of the high (left) and low (right) exposure time images. (b) The profiles of the MSE reconstructed (left) and SSIM reconstructed (right) images.

3.6 Artificial Case

For the reasons outlined in the previous subsection, an artificial case is created, to be used later in training the network proposed in this work. Because the ground truth was known, the comparison between the predictions and the ground truth become possible, as a result, the conclusion deduced from the real micro-CT case could be verified.

In CT imaging the noise consists of mainly two parts: Poisson noise arising from the fluctuations of the number of X-ray photons at the detector and Gaussian noise produced during the detector acquisition [9]. For micro-CT systems, as opposed to medical CT systems, the dominant noise is mostly characterized by Poisson distribution.

Previously obtained denoised images from the SSIM optimized network are selected as the ground truth. Then, different levels of Poisson noise are added to these images to create two datasets, in the hope that they would mimic low and high exposure time scans. In Figure 3.9, an artificially created training pair consisting of high and low quality images and the ground truth from which the images are created are given.

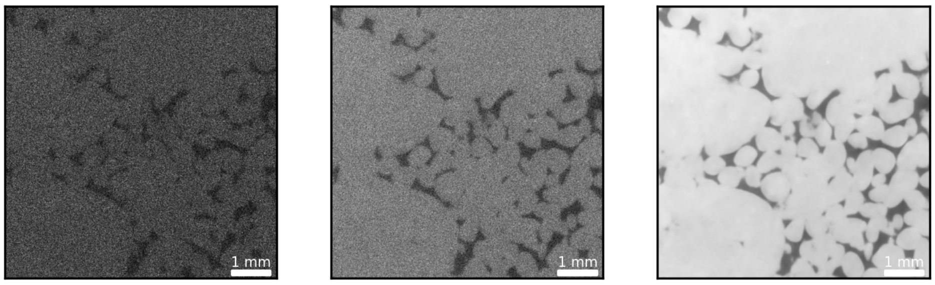


Figure 3.9: An example pair from the artificially created training data and the corresponding ground truth. From left to right: artificial low exposure image; artificial high exposure image; the reference image (ground truth).

Subsequently, the network is trained such that it learns to map from the artificial low exposure images to their corresponding high exposure ones. Similar to what has been done for the real micro-CT data previously, 80% of the data is reserved as the training set and the remaining amount is set to be the test set. The training set is further split into training and validation set in the ratio 80:20, meaning that 20% of it constitutes the validation set.

The test dataset are, then, used to assess the performance of the network against the reference images. Consequently, the quality metrics, were calculated using the reference images. Figure 3.10 depicts an example from the test set along with the network's prediction and the ground truth for this particular example. The average values for the SSIM and PSNR of the artificial high quality images are 0.42 and 27.5, respectively. On the other hand, the images denoised by the network show a great improvement, compared to the labels, reaching the average SSIM of 0.97 and PSNR of 32.5. The histogram plots for the metrics are given in Figure 3.11. Each histogram shows the frequency distribution of the quality metric values computed with respect to the ground truth. It is clearly seen from the distributions that the CNN predictions, i.e, denoised images, are closer to the ground truth in terms of the quality metrics. Hence, the results derived through the synthetic dataset testified the findings of the real micro-CT application where the ground truth was not available.

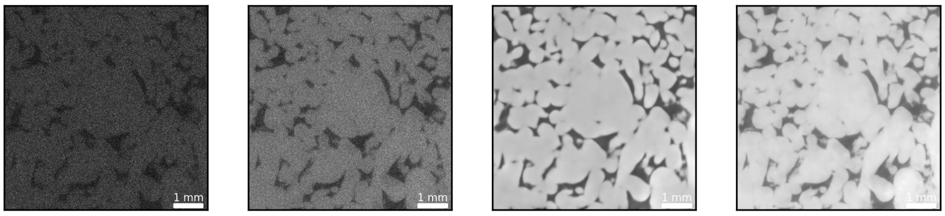
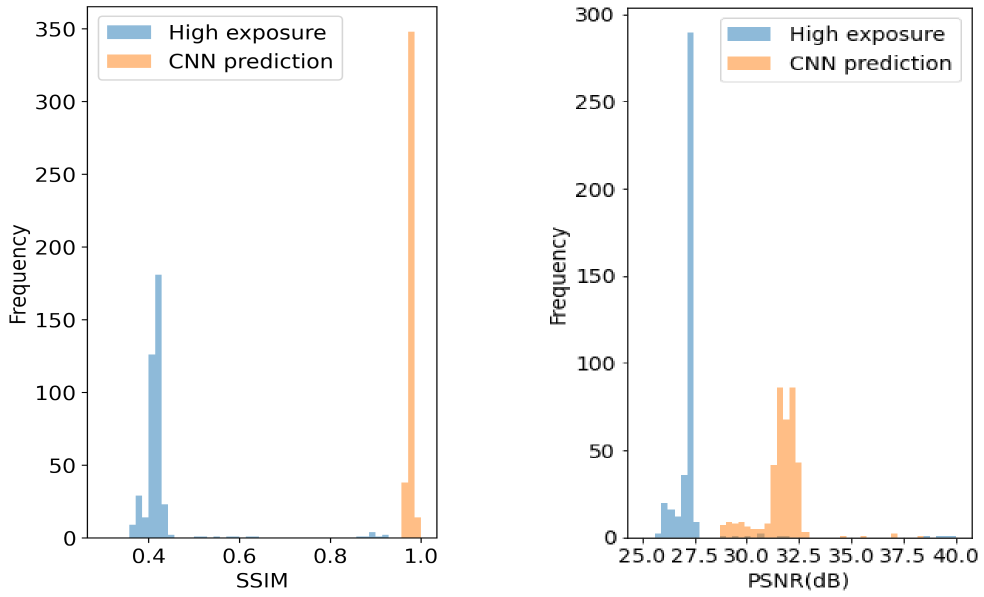


Figure 3.10: An example from the artificial test set. From left to right: artificial low exposure time image (input); artificial high exposure time image (label); the network's output (denoised image); the reference image (ground truth).



(a) (b)

Figure 3.11: Histograms of the quality metrics calculated for the artificial test set. The values are calculated with respect to the reference images (ground truth). (a) Histogram of the SSIM values. (b) Histogram of the PSNR values.

CHAPTER 4

CONCLUSION AND FUTURE WORK

CT is a powerful imaging tool exploited for different applications in several disciplines. In reservoir engineering, CT-derived digital images of rock samples allow detailed observation of microstructures of porous media and estimation of the rock properties that are vital for reservoir characterization. The quality of the tomographic images has a strong effect on the accuracy of the estimated properties. One factor that has an impact on the quality of the resulting scans is the X-ray exposure time at each projection while scanning the sample. Increased exposure times naturally yield better quality images by lowering the associated noise. Unfortunately, long scan times prevent certain processes to be captured effectively, especially when it comes to micro-CT applications. On the other hand, decreasing the exposure times leads to more noisy images, which endangers the reliability of the estimates based on these images.

As a result, several techniques have been proposed to reduce the noise associated with low-dose scans. Among them, deep learning driven approaches have gained considerable attention. They do not require prior information about the noise characteristics or the raw projection data which is hard to reach in most of the cases.

In this thesis, a U-net based DCNN is trained on a set of micro-CT scans of a carbonate rock sample to improve the quality of the reduced exposure time images. The proposed network shows a great success in reducing the exposure time more than 60% while at the same time increasing the quality. More importantly, the DCNN filter was able to distinguish between noise and other high frequency contents such as grain boundaries and sharp edges.

In this study, the network is optimized by two different loss functions, namely the MSE and SSIM in order to analyze the impact of changing the loss function. It is revealed that both functions give similar results in terms of quantitative assessment. Nevertheless, they seem to emphasize different structures of the data when we evaluate the results visually. The MSE optimized network is better at detecting fine-scale pore spaces, whereas the SSIM optimized network suggests sharper boundaries with smoother textures.

To verify the previously obtained results, an artificial case is created by assuming that the previously denoised images constitute the ground truth, which enables us to verify the results obtained from the real micro-CT dataset. The network, then, is trained on the artificially created images in a way that it maps synthetic low quality images to their corresponding synthetic high quality ones. The resulting predictions were better in quality compared to the labels according to the evaluation done with respect to the reference images (ground truth).

To sum up, in this thesis, a CNN based approach, which is completely data-driven and can be applied directly to the CT images without the need for designing a conventional filter, is shown to be successful at reducing the noise introduced by the low exposure times. This means that one can obtain good quality CT scans without requiring long acquisition times.

In CT applications, it is known that reducing the number of projections, like in the case of reducing the X-ray flux, causes less acquisition times, while, negatively affecting the image quality. Therefore, in the future, the network utilized in this work can be used for the problem of reduced number of projections. The major drawback of the method, of course, is that it requires a significant amount of data. This can be overcome by using the power of transfer learning. Another possibility is to investigate different network architectures. Generative adversarial networks are also known to be effective for image denosing tasks and they can also be employed in the area of CT imaging for reservoir rock samples.

REFERENCES

[1] H. Andrä, N. Combaret, J. Dvorkin, E. Glatt, J. Han, M. Kabel, Y. Keehm, F. Krzikalla, M. Lee, C. Madonna, M. Marsh, T. Mukerji, E. H. Saenger, R. Sain, N. Saxena, S. Ricker, A. Wiegmann, and X. Zhan, Digital rock physics benchmarks—Part I: Imaging and segmentation, Computers & Geosciences, 50, pp. 25–32, 2013.

[2] F. Baurer, M. Scrapp, and J. Szijarto, Accuracy analysis of a piece-to-piece reverse engineering workflow for a turbine foil based on multi-modal computed tomography and additive manufacturing, Precision Engineering, 60, pp. 63–75, 2019.

[3] Y. Bazaikin, B. Gurevich, S. Iglauer, T. Khachkova, D. Kolyukhin, M. Lebedev, V. Lisitsa, and G. Reshetova, Effect of CT image size and resolution on the accuracy of rock property estimates, Journal of Geophysical Research: Solid Earth, 122(5), pp. 3635–3647, 2017.

[4] M. J. Blunt, B. Bijeljic, H. Dong, O. Gharbi, S. Iglauer, P. Mostaghimi, A. Paluszny, and C. Pentland, Pore-scale imaging and modelling, Advances in Water Resources, 51(1), pp. 197–216, 2013.

[5] F. Boas and D. Fleischmann, CT artifacts: Causes and reduction techniques, Imaging in Medicine, 4, pp. 229–240, 2012.

[6] N. Buduma and N. Locascio, *Fundamentals of Deep Learning: Designing Next-generation Machine Intelligence Algorithms*, O'Reilly Media, 2017.

[7] H. Chen, Y. Zhang, M. K. Kalra, F. Lin, Y. Chen, P. Liao, J. Zhou, and G. Wang, Low-dose CT with a residual encoder-decoder convolutional neural network, IEEE Transactions on Medical Imaging, 36(12), pp. 2524–2535, 2017.

[8] H. Chen, Y. Zhang, W. Zhang, P. Liao, K. Li, J. Zhou, and G. Wang, Low-dose CT denoising with convolutional neural network, in *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*, pp. 143–146, 2017.

[9] L. Chen, L. Zheng, M. Lian, and S. Luo, A C-GAN denoising algorithm in projection domain for micro-CT, Molecular & Cellular Biomechanics, 17(2), pp. 85–92, 2020.

[10] L. De Chiffre, S. Carmignato, J.-P. Kruth, R. Schmitt, and A. Weckenmann, Industrial applications of computed tomography, CIRP Annals, 63(2), pp. 655–677, 2014.

[11] M. Diwakar and M. Kumar, A review on CT image noise and its denoising, Biomedical Signal Processing and Control, 42, pp. 73–88, 2018.

[12] G. Glatz, A. Lapene, L. M. Castanier, and A. R. Kovscek, An experimental platform for triaxial high-pressure/high-temperature testing of rocks using computed tomography, Review of Scientific Instruments, 89(4), 2018.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, The MIT Press, 2016.

[14] X. Han, J. Bian, D. R. Eaker, T. L. Kline, E. Y. Sidky, E. L. Ritman, E. L. Ritman, and X. Pan, Algorithm-enabled low-dose micro-CT imaging, IEEE transactions on medical imaging, 30(3), pp. 606–620, 2011.

[15] K. He, X. Zhang, S. Ren, and J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015.

[16] K. Hornik, M. Stinchcombe, and H. White, Multilayer feedforward networks are universal approximators, Neural Networks, 2(5), pp. 359–366, 1989.

[17] H. Ide and T. Kurita, Improvement of learning for cnn with relu activation by sparse regularization, in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2684–2691, 2017.

[18] D. Karimi, P. Deman, R. Ward, and N. Ford, A sinogram denoising algorithm for low-dose computed tomography, BMC Medical Imaging, 16(1), 2016.

[19] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, 2014.

[20] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Computation, 1(4), pp. 541–551, 1989.

[21] T. Li, X. Li, J. Wang, J. Wen, H. Lu, J. Hsieh, and Z. Liang, Nonlinear sinogram smoothing for low-dose X-ray CT, IEEE Transactions on Nuclear Science, 51(5), pp. 2505–2513, 2004.

[22] F. Mees, R. Swennen, M. Van Geet, and P. Jacobs, Applications of x-ray computed tomography in the geosciences, Geological Society, London, Special Publications, 215(1), pp. 1–6, 2003.

[23] P. Mostaghimi, M. Blunt, and B. Bijeljic, Computations of absolute permeability on micro-ct images, Mathematical Geosciences, 45(1), pp. 103–125, 2012.

[24] O. Ronneberger, P. Fischer, and T. Brox, U-Net : Convolutional networks for biomedical image segmentation, in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241, Springer International Publishing, 2015.

[25] F. Rosenblatt, The perceptron: A probabilistic graphical model for information storage and organization in the brain, Psychological Review, 65(6), pp. 386–408, 1958.

[26] D. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, Nature, 323, pp. 533–536, 1986.

[27] I. Varfolomeev, I. Yakimchuk, and I. Safonov, An application of deep neural networks for segmentation of microtomographic images of rock samples, Computers, 8(4), p. 72, 2019.

[28] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Transactions on Image Processing, 13(4), pp. 600–612, 2004.

[29] A. Weckenmann and P. Krämer, Computed tomography in quality control: chances and challenges, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 227, pp. 634–642, 2013.

[30] M. Willemink, P. A. de Jong, T. Leiner, L. M. de Heer, R. A. J. Nievelstein, R. P. J. Budde, and A. M. R. Schilham, Iterative reconstruction techniques for computed tomography Part 1: Technical principles, European Radiology, 23, pp. 1623–1631, 2013.

[31] C. You, L. Yang, Y. Zhang, and G. Wang, Low-dose CT via deep CNN with skip connection and network-in-network, in *Developments in X-Ray Tomography XII*, pp. 429–434, SPIE, 2019.

[32] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, Loss functions for image restoration with neural networks, IEEE Transactions on Computational Imaging, 3(1), pp. 47–57, 2017.