# BME 590 Final Project Report:

# Medical Image Denoising using DAE, U-Net and DnCNN with "Physical Layer" Optimization

*YueYang Pan(yp79), Xiaochen Zhou(xz237)*

**Abstract:** Medical images always come with noises, which make the later disease diagnosis harder. Denoising is an inevitable task in medical image processing. Nowadays, machine learning and deep neural network has been incorporated in image processing. However, general machine learning techniques consider mostly on the network architecture without taking the imaging system into consideration. In our project, we introduced a physical layer optimization method and added it into our deep neural networks (U-Net and DnCNN) to compete a image denoising task. Performance of the models were evaluated by their classification results of the denoised outputs.

## 1. Introduction

Usually, random noises will be generated during medical image formation. These noises on images might have negative effects in image classification and detection tasks. Traditionally, we can use median filters, histogram equalization or wavelet transform to do image denoising. As the development of machine learning techniques, more and more difficult image classification, detection, and reconstruction tasks have been efficiently solved by using (Convolutional) Neural Networks. Therefore, in this project, we are going to apply Convolutional Neural Networks (CNN) to conduct a medical image denoising task. We use three different networks: a Denoising Autoencoder (DAE), a U-Net[1], and a DnCNN[2].

## 2. Model description

### 2.1 Dataset:

The dataset we used is a Malaria dataset that has 1021 samples (cells) with labels. There are 693 non-infected samples and 328 infected samples (class 0 represents non-infected samples and class 1 represents infected samples) in total. We extracted 90% non-infected and infected samples from the dataset as our training data, while using the rest 10% samples as testing data. The size of the images is 28*28. This dataset was optimized by using a light microscope for improving the disease diagnosis accuracy[3]. The samples were illuminated by a programmable LED array which was added to the microscope. The LEDs have three spectral channels: blue, green and red (each spectral channel generates 29 channels of images). We extracted the 29 green channels as the input images in this project. The brightness of the individual LEDs can be adjusted. But as the input of our neural network, the brightness of the LEDs are fixed and we would use a "physical layer" with optimizable weights addressed to the LED patterns to determine the best arrangement and brightness of the LEDs.

Figure 1 shows the 1000th sample image in our dataset. In this image, we summed up its 29 channels and generated a one channel image as our original image (noiseless truth).

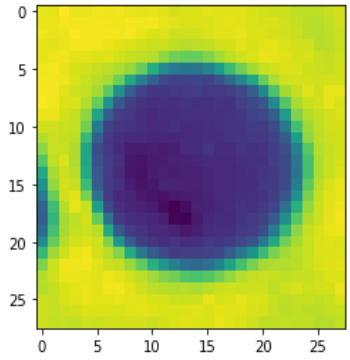


Figure1: summation of the 29 Channels (1000th sample image)

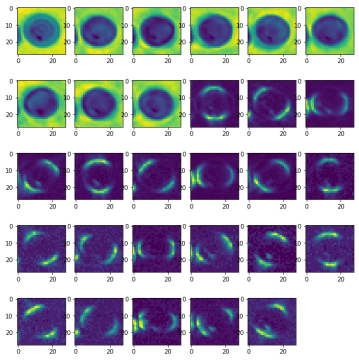

Figure 2: visualization of corresponding 29 channels

## 2.2 Physical Layer

A physical layer is added in our model to simulate the hardware setting of the microscope and will also be optimized. By training our models, we assume that we will find a set of parameters for the light microscope such that they can maximize the image denoising performance. We set an array of trainable parameters to represent the brightness of individual LEDs. To be more specific, the 29 channels mentioned in the dataset introduction part are the images generated corresponding to the 29 LED patterns. The brightness of each LED is treated as the weight of each channel. We create an array with 29 elements and each element is assigned to a channel of image as its weight. The sum of the weighted 29 channels is taken as the output image of the light microscope and as the input to our networks. To include this physical layer into our networks and make the weights trainable, we introduce a 2D convolutional layer with 29 input channels and 1 output channel. The kernel size and stride of the convolutional layer are set to be 1 and padding were set to be 0 (same padding). We also add a Sigmoid function as the activation function after the physical-convolutional layer to map the weights from 0 to 1 (0 represents the lowest brightness while 1 represents the highest brightness).

## 2.3 Model Architectures

### 2.3.1 DAE (Denoising Autoencoder)

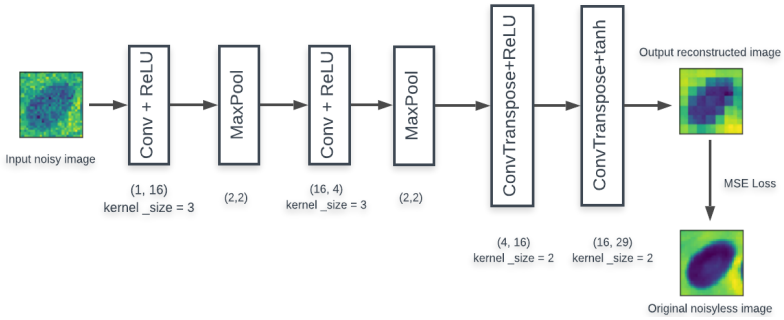


Figure 3: model architecture of DAE

An autoencoder is a type of artificial neural network used to learn efficient data codings in an unsupervised manner. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore signal "noise"[4]. The difference between a denoising autoencoder and an autoencoder is that we will use image with noises as the input of the denoising autoencoder instead of using the original images. Since the autoencoder can extract features (useful information) from input images, we expect that it can remove add-on noises of the input images after the training process. Figure 3 shows the architecture of the DAE we use.

### 2.3.2 U-Net

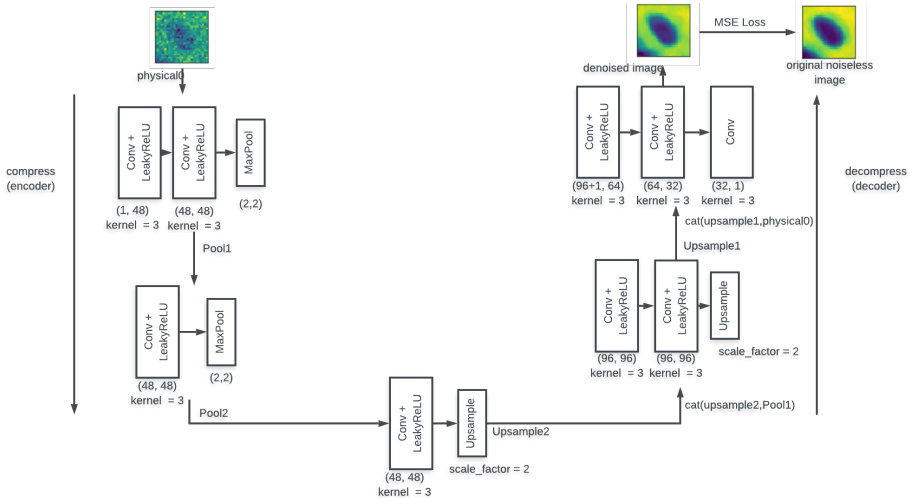


Figure 4: model architecture of U-Net

The U-Net was first developed for medical image segmentation at the Computer Science Department of the University of Freiburg, Germany[1]. It can be designed as an autoencoder which contains an encoder (compress) and a decoder (decompress). The encoder can compress image spatial features into learned filters, while the decoder can decompress learned filters back into the same spatial dimension. It was originally developed for a 512*512 image segmentation task. Since the size of the images in our project is only 28*28, we only used two blocks of the U-Net to reduce our model complexity. In the compression part, each block took an input that applied one or more 3*3 convolutional layers followed by a 2*2 max pooling. In the decompression part, each block took an input that applied multiple 3*3 convolutional layers followed by a 2*2 upsampling layer. One special property of the U-Net is: the input of a decompression layer should be concatenated with one feature map of a contraction layer. This can make sure that features that were learned by contraction will be used during reconstruction. After the encoder and decoder, the resulting features would be passing through a 3*3 convolutional layer with the desired number of output channel(s). Figure 4 shows the architecture of the U-Net we use for this project (with notes of some specific information, such as the number of input/output channels and kernel size, etc.).

### 2.3.3 DnCNN (Denoising Convolutional Neural Network)

Denoising convolutional neural network (DnCNN) benefits from the integration of two strategies: residual learning and batch normalization. Residual learning was first introduced to solve the performance degradation problem[5]. In DnCNN, a noisy image will be given and the model will be trained to remove latent clean image from the noisy one. The output of DnCNN will be the predicted residual image. Widely used to improve the training efficiency, batch normalization reduced the covariate shift during the training process and is also employed[2]. It has been empirically shown that the integration of residual learning and batch normalization can benefit from each other and resulted in faster training speed as well as better denoising performance. Also, DnCNN has shown the potential of general image denoising, such as blind noise level denoising.

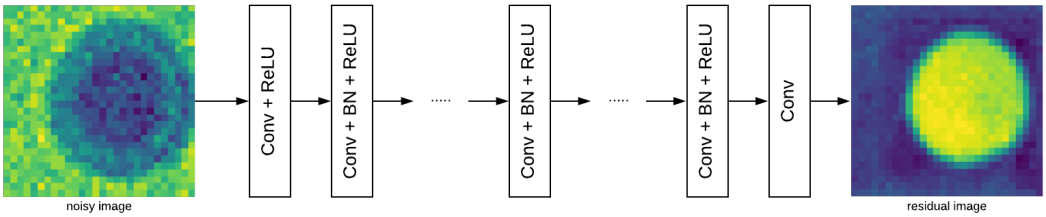


Figure 5: model architecture of DnCNN

The depth of the network is selected based on the noise level. A deeper network results in a wider receptive field and can capture more context information, which helps the denoising task for images with high noise levels. Since we introduce random noises with standard deviation up to 0.2 (Gaussian or Poisson, unknown in the training process), we set the depth of DnCNN to be 20.

## 3. Experiment Results and Discussion

Figure 6 shows the original images, noisy images (mean: 0; standard deviation: 0.05) and the denoised images reconstructed by DAE. The DAE provides a very blur denoising results. The reason why the results of DAE are not convincing might be: our DAE model does not have enough complexity and does not have upsampling steps.

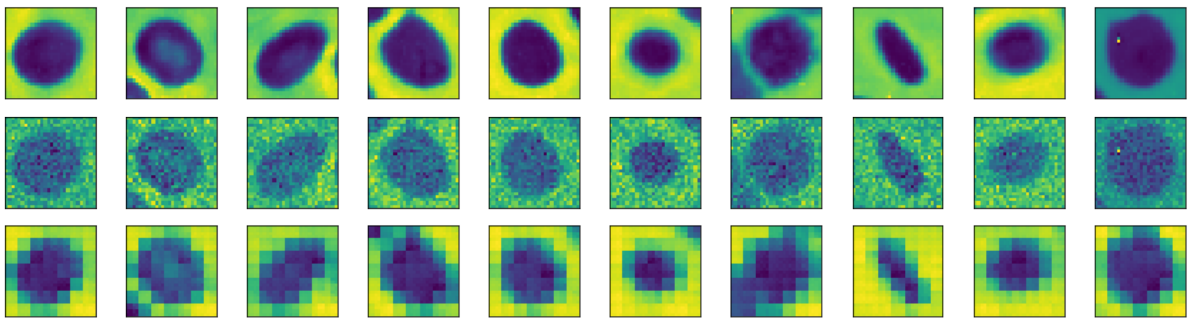

Figure 6 The Denoising Results of DAE

Since the denoising results of DAE are not convincing, we move to the other two network: U-Net and DnCNN, to see whether there is an improvement. For the later two networks (U-Net and DnCNN), the inputs for the networks are the original images with a random generated gaussian or poisson noise (mean: 0, standard deviation: 0.2). First, we train a U-Net without a physical layer optimization. Figure 7 and figure 8 shows the training and testing results of our non-physical optimized U-Net. The images in the first row are the original noiseless images; the images in the second row are the noisy images; the images in the third row are the denoised results provided by our U-Net after 50 epochs training. Then, we train a U-Net with the physical layer optimization (50 epochs). Figure 9 and figure 10 shows the training and testing results of our physical optimized U-Net. The results are shown as below:

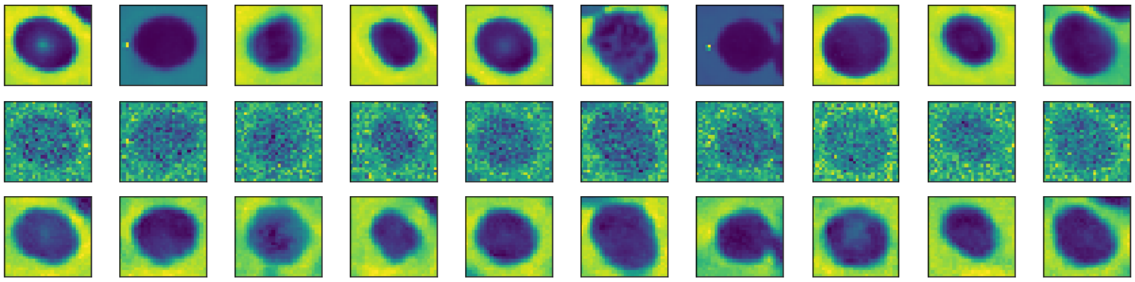

Figure 7 The Denoised Images of the Non-physical Optimized U-Net (training results)

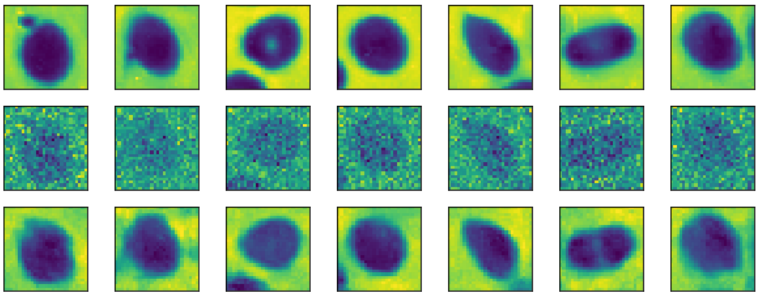

Figure 8 The Denoised Images of the Non-physical Optimized U-Net (testing results)

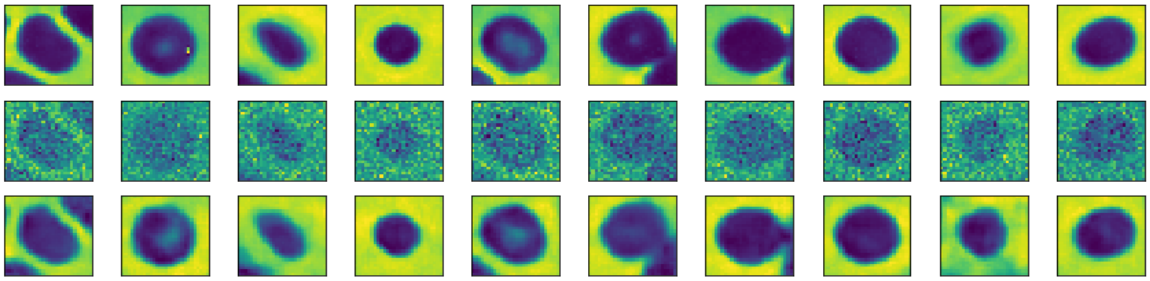

Figure 9 The Denoised Images of the Physical Optimized U-Net (training results)

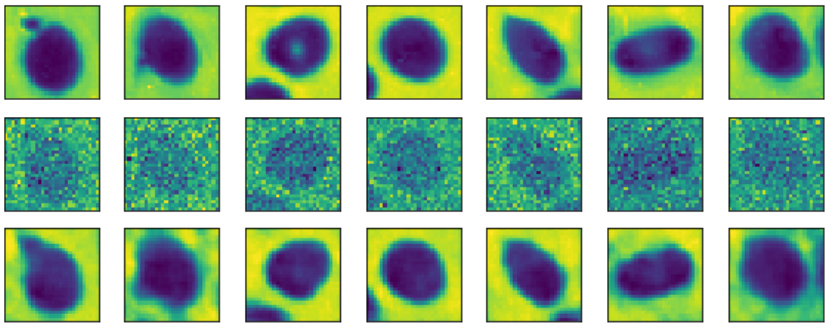

Figure 10 The Denoised Images of the Physical Optimized U-Net (testing results)

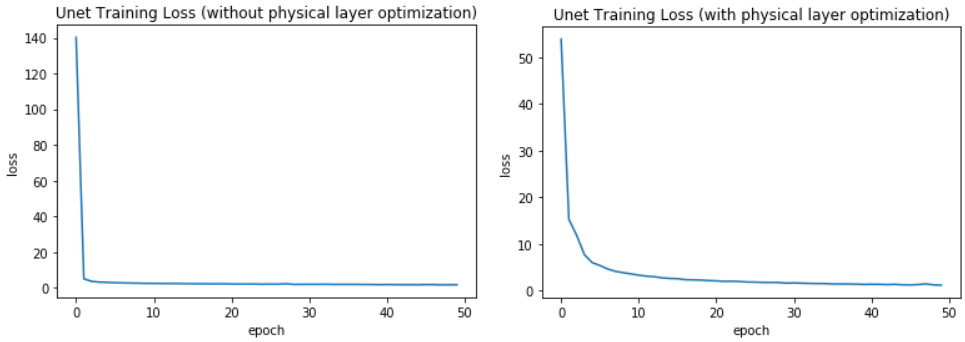


Figure 11 Training Loss of Non-physical optimized U-Net and Physical optimized U-Net

From the above results of the Non-physical Optimized U-Net and the Physical Optimized U-Net, we can observe that the training loss of the U-Net without physical optimization converges faster than the training loss of the one with physical optimization. However, according to the training loss from their last epoch, the Physical Optimized U-Net (mse = 1.18) has a lower MSE loss than the Non-physical Optimized U-Net's (mse = 1.83). And from the above denoising results, the Physical Optimized U-Net seems having a better denoising performance (with relatively clear edges and details) than the Non-physical Optimized one. We also trained DnCNN with and without physical layer optimization.

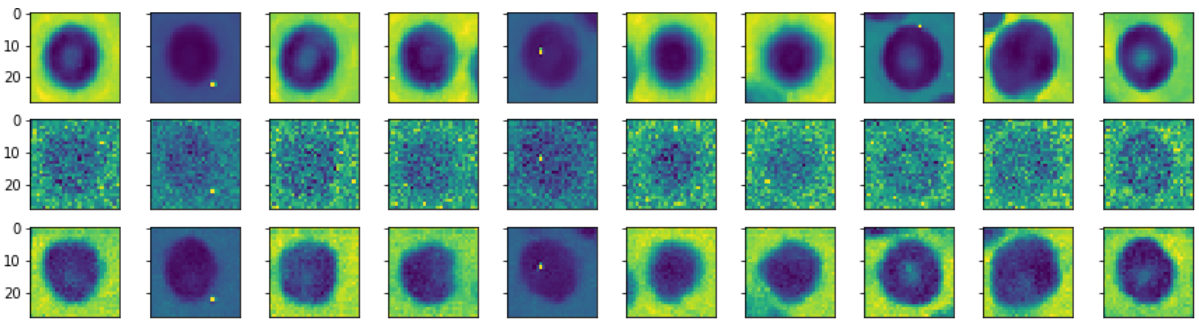

figure 12 The Denoised Images of the non-physical Optimized DnCNN (train results)

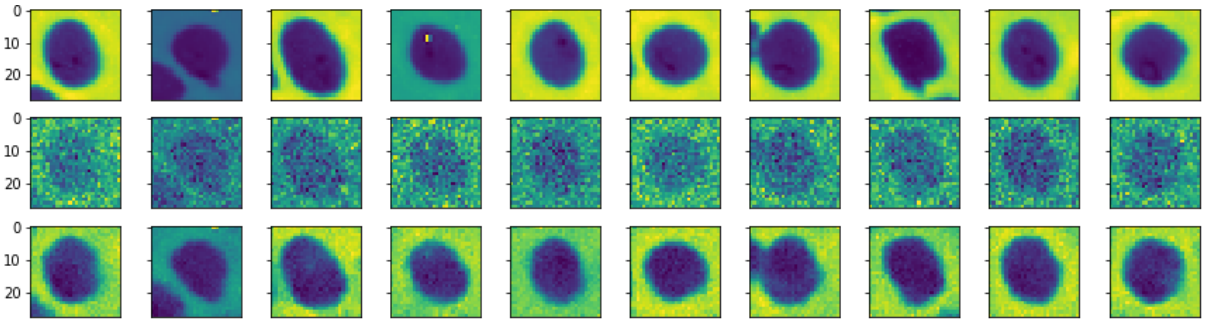

figure 13 The Denoised Images of the non-physical Optimized DnCNN (test results)

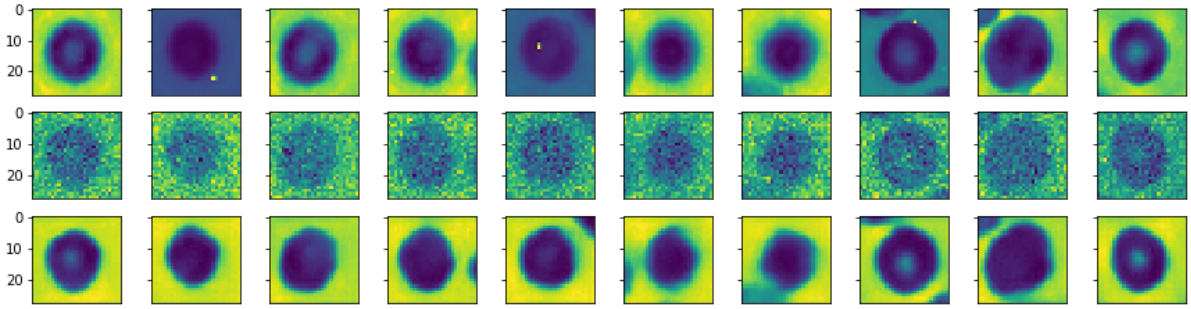

figure 14 The Denoised Images of the physical Optimized DnCNN (train results)

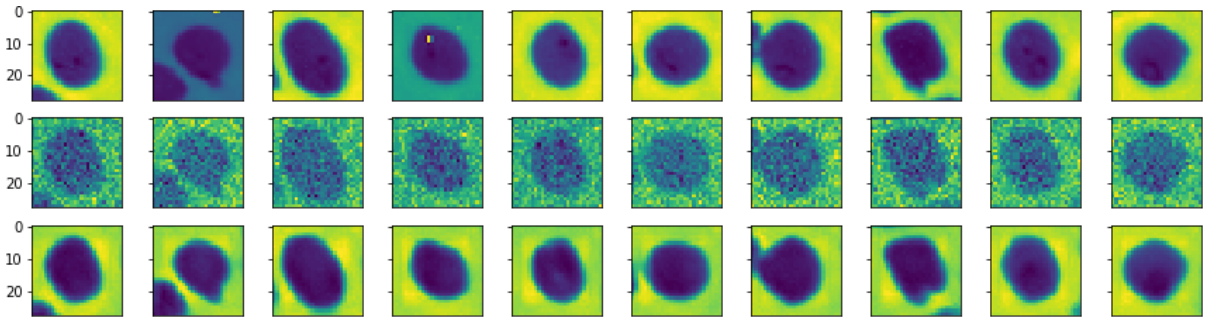

figure 15 The Denoised Images of the physical Optimized DnCNN (test results)

As can be clearly seen from the results above, the training process of the physical layer does improve the denoising performance for both train and test data. The results show that denoised images of non-physical model still contains visible noisy pixels and lose most features contained in the original noiseless images. The physical model, however, perform much better with most features reserved.

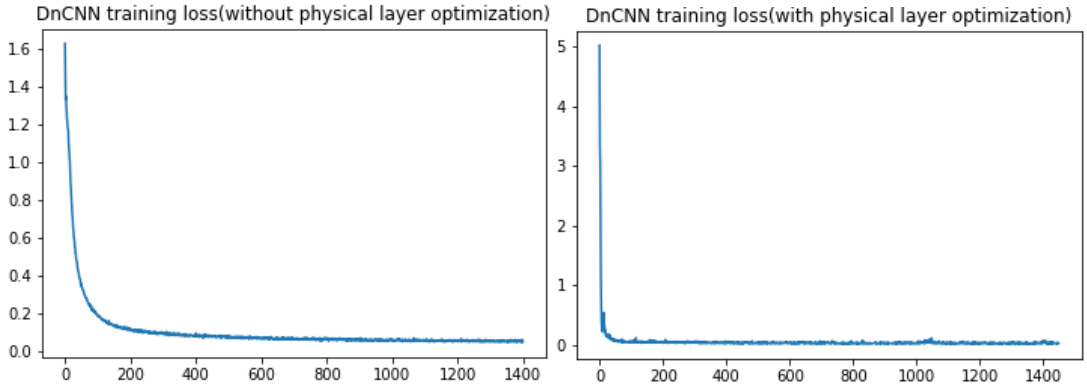


Figure 16 Training Loss of Non-physical optimized DnCNN and Physical optimized DnCNN

To better observe the network's capacity and convergence performance, we train the models with and without physical optimization for 50 epochs. We can see from Figure 16 that the loss of physical optimized model converges much faster than the non-physical one. By introducing the physical layer, we've both boosted the denoising performance and saved time for reaching the convergence.

| MSE Loss | U-Net | DnCNN |
|---|---|---|
| Non-Physical Optimized Training Loss (last epoch) | 1.8307 | 0.0529 |
| Physical Optimized Training Loss (last epoch) | 1.1830 | 0.0332 |
| Non-Physical Optimized Test Loss | 1.6021 | 0.0454 |
| Physical Optimized Test Loss | 1.4107 | 0.0447 |

Table1: train/test loss for U-Net and DnCNN

We also report the training and testing MSE loss for U-Net and DnCNN for each model (the MSE can be used to indicate the denoising performance). As we can see from the results below, for both U-Net and DnCNN, models with physical layer optimization perform better than those without physical layer optimization (conforms to our expectation). Note that even the MSE Loss of U-Net is greater than the MSE of DnCNN, that doesn't mean DnCNN performs better than U-Net. Actually, according to figure 7-15, we cannot tell a difference between the training and testing denoised images from both of the networks. That's because MSE doesn't consider the structure similarity between two images.

To further evaluate our models, we also use the original images, noisy image and the models' output (denoised images from each model) for classification. To clarify the difference between the models, we train a simple CNN classifier for each type of the images (original, noisy and denoised) for 50 epochs. In this classification task, we calculate the cross-entropy loss of our predicted classes and the truth labels for backpropagation. The training accuracy is calculated by using the predictions and the truth from the last batch of the last training epoch. The

testing accuracy is calculated by using all the test images with their corresponding labels. The classification results are shown as below:

| | Original Image | Noisy Image | Non-Physical U-Net | Physical U-Net | Non-PhysicalDnCNN | Physical DnCNN |
|---|---|---|---|---|---|---|
| Training Accuracy | 0.86 | 0.75 | 0.63 | 0.81 | 0.72 | 0.72 |
| Testing Accuracy | 0.73 | 0.68 | 0.68 | 0.70 | 0.72 | 0.74 |
| Training Loss | 0.46 | 0.51 | 0.63 | 0.49 | 0.62 | 0.59 |
| Testing Loss | 0.46 | 0.52 | 0.63 | 0.50 | 0.36 | 0.48 |

Table 2: CNN classification results for original, noisy and denoised images

We can observe from table 2 that the training and testing accuracy of the classification of Physical Optimized U-Net denoising results are close to the accuracy of classification using original images. And the performance is much better than the classification of noisy images and denoised images provided by the Non-Physical Optimized U-Net. For DnCNN, the training accuracy of Physical and Non-Physical Optimized model is similar, but the Physical Optimized model performs better in testing. They both perform better in testing compared with noisy image. One possible explanation of Noisy image's relatively better training performance is overfitting.

In conclusion, for both U-Net and DnCNN, the model with physical optimization performs better than the one without physical optimization. This result proves that by adding a physical layer to adjust the brightness of the individual LEDs (strengthen different features of the cell images) can improve the model's denoising performance and provide a better classification result.

## 4. Conclusion

In our project, we introduced physical layer optimization into some well-known denoising networks. Except of the denoising autoencoder, which doesn't perform well itself, our U-Net and DnCNN model with physical layer optimization performs better than when the model without physical layer optimization. We used a Malaria dataset and simulated the brightness adjusting process of the LEDs in the light microscope by training the weights of 29 channels images (correspond to 29 different LEDs). We reported both the denoised image and the classification results for U-Net and DnCNN with physical layer optimization and compared them with the results from original U-Net and DnCNN without physical layer. Our model performed better both in denoised image's visual quality and the classification result, which indicates that by simulating a physical layer and train it along with some well-structured denoising models, we are able to find a set of parameters to optimize the light microscope (adding features to the images) that can help future classification or denoising tasks.

### References

[1]Olaf Ronneberger, Philipp Fischer, Thomas Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", arXiv e-prints, arXiv:1505.04597, May. 2015.
[2] Zhang, Kai et al. "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising." IEEE Transactions on Image Processing 26.7 (2017): 3142–3155. Crossref. Web.
[3] Alex Muthumbi, Amey Chaware, Kanghyun Kim, Kevin C. Zhou, et al., "Learned sensing: jointly optimized microscope hardware for accurate image classification", Biomedical Optics Express, Vol. 10, pp. 6351-6369 (2019)
[4] Goodfellow, Ian; Bengio, Yoshua; Courville, Aaron (2016). Deep Learning. MIT Press. ISBN 978-0262035613
[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.