Mini-project 4

Exploring denoising for Content-aware image restoration

Group 31

Ketan Rampurkar (260732873)

Nick Chahley (260865097)

Saifullah ElSayed (260733168)

Abstract

Fluorescence microscopy is the stepping stone towards numerous discoveries and breakthroughs in life-sciences. With the optics of the microscope, chemistery of the fluorophores,and maximum photon exposure tolerated by sample being limitations of fluorescence microscopy, multiple trade-offs between imaging speed, spatial resolution, light exposure and imaging depth. In this paper we show how image restoration based on deep learning extends the range of biological phenomena observable by microscopy. On two concrete examples of the research paper **Content-aware image restoration: pushing the limits of fluorescence microscopy** [1], we are reproducing the results of one of the image restoration processes which is denoising. We then assess the effects of changing the hyperparameters values of the networks and investigate their generalizability to datasets that differ from those which they were trained on.

1 Introduction

The absorption and subsequent re-radiation of light by organic as well as inorganic specimens is typically the result of well-established physical phenomena described as being either fluorescence or phosphorescence. The emission of light through the fluorescence process is nearly simultaneous with the absorbtion of the excitation light due to a relatively short time delay between photon absorption and emission, ranging usually less than a microsecond duration. When emission persists longer after the excitation light has been extinguished, the phenomenon is referred to as phosphorescence.

The technique of fluorescence microscopy has become an essential tool in biology and the biomedical sciences, as well as in materials science due to attributes that are not readily available in other contrast modes with traditional optical microscopy. The basic function of a flourescence microscope is to irradiate wavelength-specific fluorophores within the specimen and then capture their emitted fluorescence with a detector while filtering out light from the excitation source. In properly configured microscope, only the emission light should reach the detector so that the resulting fluorescent structures are superimposed with high contrast against a black bacground. The limits of detection are generally governed by the darkness of the background, and the excitation light is typically several hundred thousand to a million times brighter than the emitted fluorescence.

Due to recent advances, fluorescence microscopy enables time-resolved volumetric of biological processes within cells at high resolution. However, the quality of these processes is determined by spatial resolution of optical device, desired temporal resolution, total duration of the experiment, the required imaging depth, the achievable fluorophore density, bleaching, and photo-toxicity. For the optimizations of these processes, trade-offs are put into place , since its impossible to optimize all the processes at the same time. An example of these trade-offs is sacrificing signal-to-noise ratio (SNR) by reducing exposure time to gain imaging speed. These trade-offs can be addressed through the optimization of the microscopy hardware, yet there are physical limits that cannot easily be overcome. Therefore, computational procedures to improve the quality of acquired microscopy images are becoming increasingly important. Here is where deep learning comes in.

The direct application of deep learning methods to image restoration tasks in fluorescence microscopy is complicated by the specific modality and sources of error that come with imaging live specimens at a small scale. Fortunately, the 2018 paper **Content-aware image restoration: pushing the limits of fluorescence microscopy** [1] has presented a deep learning toolbox specficically tailored for use with fluorescence microscopy of biological samples on a variety of tasks.

1.1 Task Description

In this paper, we will be focusing on one of the image restoration tasks, which is image denoising. A particular thing to note is that our training model was trained for a specific combination of image content(e.g. nuclei, microtubules) and image corruption (camera noise, pixel size, microscope PSF, etc.). In the following sections

we will be discussing our approach for reproducing a subset of the results presented in [1]. Specifically, the two restoration experiments with physically acquired training data (denoising of planaria and tribolium). Moreover, we will be comparing the image metrics and how the models of these two experiments differ and vary by changing their given parameters, and observing the change of image restoration quality. We will examine how the runtime and performance of the CARE network is affected by hyperparameter settings (batch size, steps per epoch). Finally, we will perform a generalization experiment in which we test the models on different datasets (in terms of specimen, exposure conditions, microscope) than they were trained on.

2 Related Work

Numerous and diverse denoising methods have been proposed in the past decades- most of these methods estimate the denoised pixel value based on the information provided in the surrounding local limited window. Unlike these local denoising methods, non-local methods estimate the noisy pixel based on the information of the whole image. Buades[2] developed a non-local image denoising algorithm by making use of the information encode in the whole image. When modifying a pixel, the algorithm first computes the similarity between a fixed window centered around it and the windows centered around the other pixels in the whole image, then it takes the similarity as a weight to adjust this pixel. While this method shows remarkable and convincing results and has been implemented in the paper we are replicating, the efficiency was observed to be low for its pixel-wise window matching. The computational complexity of the NLM algorithm is about $O(n^4)$, which made it unfeasible for us to train our model, as shown in the training time of the model in the paper we are replicating.

A much faster approach for biomedical image processing is using deep convolutional networks. Our CNN is based on U-Net [3], which is an encoder-decoder architecture with skip connections. In contrast to most CNNs for image restoration, we don't predict a single value per pixel, but instead a distribution. In particular, we predict the location μ and scale σ of a Laplace probability density function. By reducing the spatial resolution of internal feature layers with pooling layers, it leads to better image features. Using upsampling to successively enlarge feature layers in the decoder will eventually lead to an output that is of the same resolution as the input. Section 4.3 gives a more detailed insight of the U-Net architecture implemented in CARE.

3 Dataset

We set out to replicate two restoration experiments from the paper: denoising planaria and denoising tribolium [1]. While both datasets were developed for the task of denoising 3D volumes acquired from confocal microscopy, they differ in terms of organism imaged, sets of illumination/exposure conditions making up the

high and low signal-to-noise ratio (SNR) groups, and type of microscope used. We obtained these datasets from the link provided by the publication [4].

3.1 Planaria

The data consists of four groups: a high SNR condition which contains the target, ground-truth (GT) images, and three low SNR conditions containing inputs to be restored and compared to the GT. Images are of fixed, RedDot1-labeled flatworm (*Schmidtea mediterranea*) acquired by spinning disk confocal microscopy. The training data contains 17,900 16x64x64 (ZYX) sub-volume stacks sampled from 96 confocal stacks of average size 400x1024x1024 stored as numpy arrays. We performed a 80:20 training:validation split. The test data contains 20 previously unseen Z-stacks of 1024x1024 images for each of the four conditions.

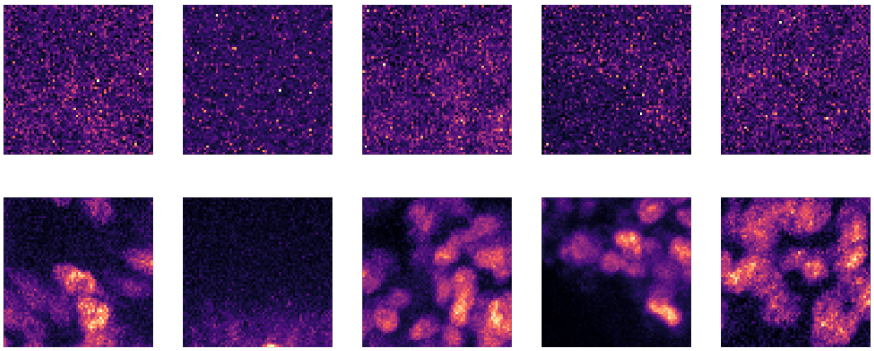

Figure 1: Example validation patches from the planaria dataset. Top row is inputs and the bottom row is their corresponding targets (ground truth).

3.2 Tribolium

As with the planaria dataset, the tribolium data consists of four groups: a high SNR condition which contains the target, ground-truth (GT) images, and three low SNR conditions containing inputs to be restored and compared to the GT. Images are of RedDot1-labeled red flour beetle (*Tribolium castaneum*) specimes acquired by pinhole confocal microscopy. The training data contains a total of 15,500 sub-volumes of the same shape (16x64x64) as the planaria set sampled from 26 full image volumes and was split 80:20 into training and validation sets. The test dataset is 6 reserved ZYX volumes.

4 Approach

4.1 Software

Code for training and prediction with the CARE networks used in our selected paper is provided in the CBSDeep Python library, available on PyPI. This is a Tensorflow-based toolbox implementing a base CNN

class, CARE, for image denoising, and extensions of that network for specialized restoration tasks (*e.g., IsotropicCARE* for isotropic reconstruction). CSBDeep also contains a number of utility functions for image normalization, dataset construction, and model training.

4.2 Image Comparison Metrics

Our selection image normalization and comparison methods was informed by the article's supplementary notes (*2.2 Preliminaries*) [1]. To give us the best chance of replicating the results of our chosen paper, we used functions implemented in the CSBDeep package over other python image libraries whenever possible.

For training and evaluation we normalize the input images using a percentile based method (csbdeep.utils normalize), defined for an image u as

$$N(u; p_{low}, p_{high}) = \frac{u - perc(u, p_{low})}{perc(u, p_{high}) - perc(u, p_{low})} \tag{1}$$

where $perc(u, p)$ is the p-th percentile of the pixel intensities of u. For input normalization we used values of $p_{low} = 3$ and $p_{high} = 99.8$. Additional normalization is carried out on the restored image prior to computing the comparison metrics. First the ground truth image y is normalized using Eq. (1) with $N(u; 0.1, 99.9)$. Then we apply an affine rescaling to the restored image $\hat{y}$ using the linear transformation $\phi(\hat{y}) = \alpha\hat{y} + \beta$ which minimizes the MSE between $N(y; 0.1, 99.9)$ and $\phi(\hat{y})$ (csbdeep.util.normalize_minmse).

To evaluate the quality of image restoration we computed two metrics. The *normalized root-mean-square error (NRMSE)*, defined as

$$NRMSE(y, \hat{y}) = \sqrt{MSE(\phi(\hat(y), N(y, 0.1, 99.9)} \tag{2}$$

is a measure of the overall difference in pixel values between the two images (lower is better). The *structural similarity index (SSIM)* [5] is a measure of the perceived similarity between two images (higher is better) which parameterizes comparisons of luminosity, contrast, and structure. We used scikit-image to compute SSIM (skimage.measure.compare_ssim).

4.3 CARE Architecture

U-Net is a class of convolutional networks developed for use in biological imaging applications. The CARE network architecture is based on a 3D U-Net customized for a probabilistic model, that predicts a per-pixel distribution (parameterized by mean and scale) resulting in a 2-channel output. The mean is calculated as the sum of the input and the internal predictions (residual addition layer near the end). For the non-probabilistic model only the mean is used as prediction.

The network architecture consists of a contracting path and an expansive path. The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 con-

volutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling. At each downsampling step we double the number of feature channels. Every step in the expansive path consists of an upsampling of the feature map followed by a 2x2 convolution ("up-convolution") that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution. At the final layer a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes. In total the network has 23 convolutional layers.[6]

We selected the parameters of our CARE networks to match those reported by the authors in supplementary table 3 [1]. All our models have 2 pooling layers, a filter of size 3, and 32 initial convolutional features after the first max pooling layer. Unless otherwise stated (as in the hyperparameter experiments) models were trained with hyperparameters are as follows: batch size 16, 400 parameter update steps per epoch, and a progressive learning rate initialized at 0.0004 and set to change by a factor of 0.5 after 10 epochs at plateau. Training was to 200 epochs in the replication experiments and to 100 epochs the subsequent hyperparameter and generalization experiments.

5 Results

5.1 Replication

Our planaria models improved the RMSE and SSIM scores (Table 1) of the inputs by a higher margin than those reported in the original paper (*e.g.,* SSIM improvement of 8-18 times vs 3-4 times). In contrast our tribolium models scored lower in both relative and absolute accuracy than in the original study. For both the planaria and tribolium datasets, the metrics we calculated for our input images rate them as less similar to the GT than those of the original paper. Our models had faster training times (200 epochs) on both datasets (planaria: 7h57m vs 30h06m; tribolium: 7h51m vs 26h12m). Both us and the authors performed these computations on a single Nvidia 1080 GPU ([1], supplementary table 3).

	RMSE				SSIM			
	Replication		Original		Replication		Original	
Condition	Input	Network	Input	Network	Input	Network	Input	Network
C1	0.19068	0.02666	0.08384	0.03768	0.10476	0.91439	0.21176	0.70774
C2	0.23019	0.03133	0.08818	0.03942	0.07513	0.89586	0.17071	0.70803
C3	0.30839	0.04347	0.09103	0.0478	0.04263	0.79344	0.14415	0.60972

Table 1: Restoration accuracy of our best planaria model on the *Schmidtea mediterranea* test dataset contrasted with the values in the original paper. Condition 1 has the highest SNR, condition 3 the lowest.

	RMSE				SSIM			
	Replication		Original		Replication		Original	
Condition	Input	Network	Input	Network	Input	Network	Input	Network
C1	0.16192	0.09671	0.09556	0.03301	0.07729	0.33225	0.17048	0.88278
C2	0.1284	0.08481	0.0891	0.02751	0.12726	0.45318	0.22931	0.9067
C3	0.09521	0.07256	0.07629	0.02356	0.23354	0.61392	0.36174	0.92217

Table 2: Restoration accuracy of our best tribolium model on the *Tribolium castaneum* test dataset contrasted with the values in the original paper. Condition 1 has the lowest SNR, condition 3 the highest.

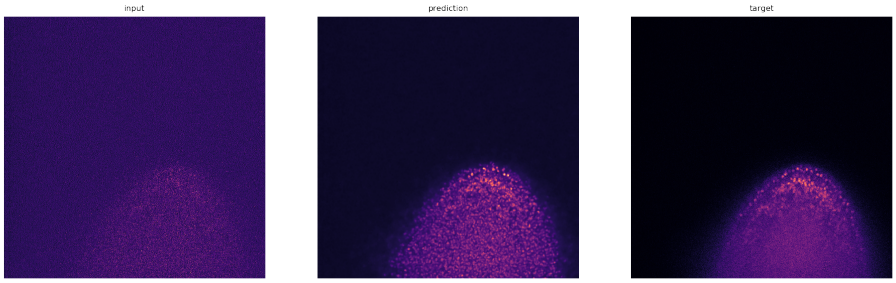

Figure 2: An image of *Schmidtea mediterranea* from the planaria test data restored by our CARE network (prediction) restored from a low SNR image (input) corresponding to a high SNR ground truth (target).

5.2 Hyperparameter Study

We assessed the effect of changing the hyperparameters batch size and parameter update steps per epoch on the training time and restoration accuracy of CARE networks trained on the *Schmidtea mediterranea* (planaria) dataset.

Changing the training batch size from the value used in the paper (16) did little to affect restoration accuracy (Fig. 3). An exception is found in the lowest SNR condition (3), where the SSIM of the model with batch size 8 (half of the default) drops more than the others. Runtime was positively correlated with batch size. The time to train a model to 100 epochs increased by an average factor of 1.8 with each doubling of its batch size: 2h23m (8), 4h00m (16), 7h25m (32), 14h28m (64).

CARE models with the highest step number performed the best in conditions 1 and 2. In condition 3 the 100 step model outperforms its 400 step cousin. As with batch size, the time to train to 100 epochs was positively correlated with the number of parameter updates per epoch: 0h51m (25), 1h29m (100), 4h00m (400).

5.3 Generalizability Study

When tested on the planaria data, the tribolium model underperformed the planaria by 20-22% on SSIM and 12-18% on RMSE. When the same models were tested on the tribolium data, the planaria trained model performed comparable to the tribolium model (differences within 1%).

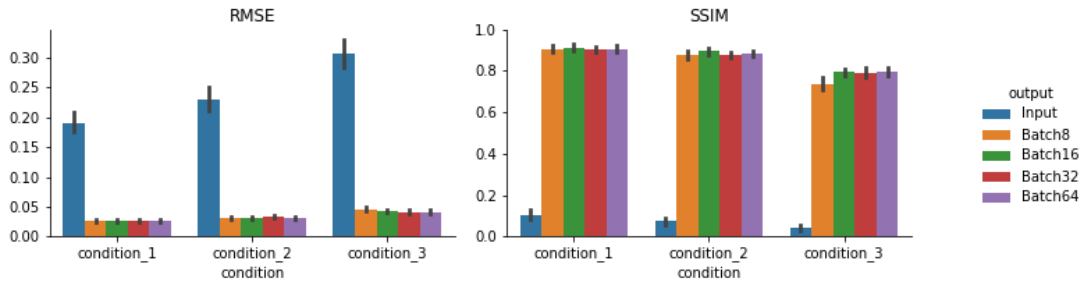

Figure 3: Effect of batch size on the restoration accuracy of planaria and tribolium-trained models across different test datasets.

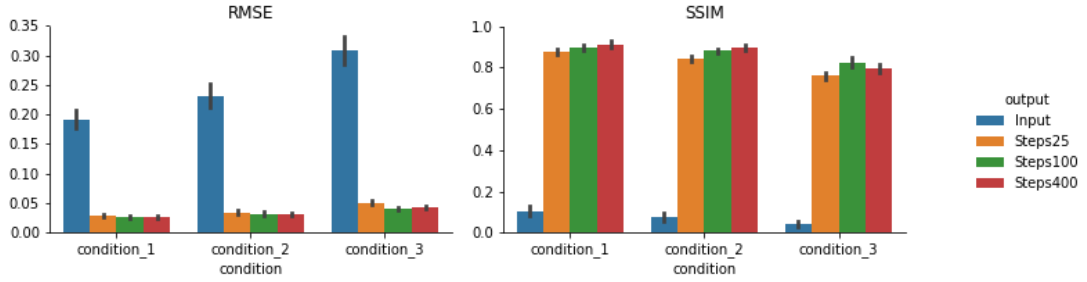

Figure 4: Effect of batch size on restoration accuracy for CARE models trained on the planaria dataset. The default batch size reported in the original paper is 16.

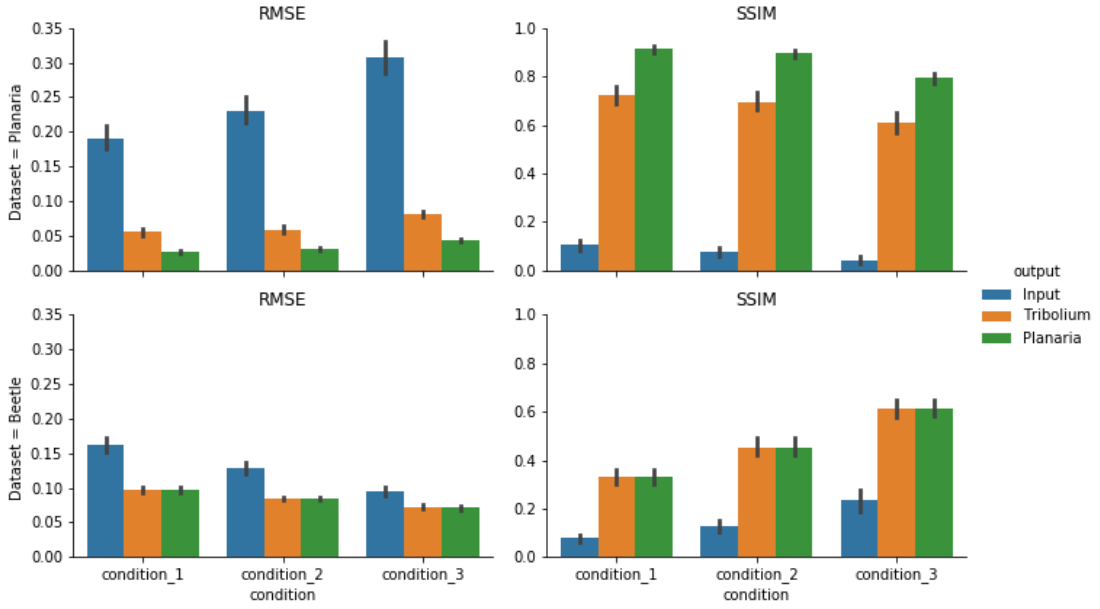

Figure 5: Effect of the number of parameter update steps per epoch on restoration accuracy for CARE models trained on the planaria dataset. Our default CARE models use a value of 400. The original paper does not cite which value was used for the steps parameter, but the CSBDeep documentation strongly encourages 400 steps per epoch for a production model.

6 Discussion and Conclusion

In terms of replication of the results, there is a discrepancy between our metrics observed and the ones reported in the existing paper. SSIM scores varied mostly because of the differences in its implementation. We also see differences in the runtime of the 2 models, explained mainly by the difference in one of the hyperparameters- steps per epoch. There was no explicit mention of the number of epochs used in the paper. However, a simple linear regression to extrapolate our data from runtime vs n steps per epochs for the planaria data concluded with around 1800 steps per epoch for the runtime reported by the authors- which is 10 times more than our steps per epoch. This justified the vast difference in the training time for the 2 models.

We noticed that changing the hyperparameters did not increase the accuracy of the model by much- only the runtime was affected. Although dropping below the default values (batch 8, steps per epoch 25) did lower performance on the low SNR conditions (C3) a bit - which suggests that the default parameters suggested in the paper are a good compromise between speed and performance of the model.

For the generalizability study, we concluded that both the models performed better than expected. Given that CARE models were explicitly supposed to be trained on a specific species, set of exposure conditions, and microscope, all of these assumptions were violated when we decided to test our training model on a different test dataset. Since the tribolium model underperformed the planaria, we concluded that the only difference between the 2 models were the datasets provided- the planaria dataset is just as good at training for the tribolium data while that's not the case for planaria.

6.1 Code Availability

The code for CARE network training and prediction (in Python/TensorFlow) is publicly available at
https://github.com/ksr987/COMP551-MP4

7 Statement of Contributions

We all researched past work related to fluorescence microscopy and brainstormed an appropiate approach. Nick's main task was implementation, training and testing of the code. Ketan's main task was writing the report with the help of Saifullah. All members assisted each other in these main tasks and were in constant communication to ensure cohesion throughout.

References

1. Weigert, M. *et al.* Content-Aware Image Restoration: Pushing the Limits of Fluorescence Microscopy. *Nature Methods* **15,** 1090. ISSN: 1548-7105 (Dec. 2018).

2. Rudin, L. I. & Osher, S. *Total variation based image restoration with free local constraints* in *Proceedings of 1st International Conference on Image Processing* **1** (1994), 31–35.

3. Ronneberger, O., Fischer, P. & Brox, T. *U-net: Convolutional networks for biomedical image segmentation* in *International Conference on Medical image computing and computer-assisted intervention* (2015), 234–241.

4. Weigert, M. *et al. Content Aware Image Restoration: Pushing the Limits of Fluorescence Microscopy. Supplementary Data* https://publications.mpi-cbg.de/publications-sites/7207/ (2019).

5. Wang, Z., Bovik, A., Sheikh, H. & Simoncelli, E. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing* **13,** 600–612. ISSN: 1057-7149 (Apr. 2004).

6. Ronneberger, O., Fischer, P. & Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. http://arxiv.org/abs/1505.04597 (2019) (May 18, 2015).