Machine Learning Based 3D XCT Image Enhancement for the Inspection of AM Metal Components using Limited X-ray Measurements

Thomas Blumensath[a], Ibrahim Harrane[a]

[a]*μ-Vis X-ray Imaging Centre, University of Southampton, University Road, Southampton, SO17 1BJ, UK*

Abstract

The determination of the size and density of internal defects is a crucial step in the inspection of additively manufactured metal components. X-ray tomographic imaging is often the only viable inspection techniques, but for objects with complex shapes, imaging can be difficult, leading to images with significant artefacts that mask internal defects. Data-driven machine learning based image enhancement tools have demonstrated benefits in this setting, however, current methods often rely on the availability of significant training data. In this paper we evaluate methods that are trained on a single volumetric image. We are particularly interested in imaging applications where only limited X-ray measurements can be collected. In this setting, we compare a range of network architectures and training approaches and show how block based image processing can have significant benefits. Using both simulated and real X-ray data, we show that block based models can be trained efficiently on single volumetric images. Comparing several deep learning structures, we find that a 3D U-net architecture, applied to small image blocks outperforms a wide range of alternative approaches in this setting, showing particular benefits in challenging applications where X-ray images have significant artefacts.

Keywords: X-ray tomography, additive manufacturing, limited measurements, machine learning, limited training data

Email address: thomas.blumensath@soton.ac.uk (Thomas Blumensath)

1. Introduction

The non-destructive inspection of manufactured components is a vital step in the quality assurance process and is indispensable in many safety-critical applications. Advances in manufacturing processes and particularly modern additive manufacturing (AM) methods, now allow us to build components with complex shapes and advanced internal structures. Whilst these can lead to components with significantly improved performance, non-destructive inspection often remains challenging, as current techniques are often not directly applicable. We are here interested in the inspection of components manufactured using metallic powder bed fusion (PBF) additive manufacturing (AM). Material feedstock irregularities, powder recoating, part solidification and material/laser interactions lead to process parameter dependent defects that reduce mechanical properties [1, 2, 3, 4, 5]. We are thus interested in characterising the pore structure of metal parts manufactured using a laser-based PBF process. This pore structure is known to be affected by a wide range of manufacturing parameters and will have a significant impact on structural component performance [6]. X-ray tomography [7] offers a powerful approach to the inspection of such components, often allowing a detailed visualisation of micrometer structures throughout the object. However, the image resolution directly depends on the object size. Furthermore, especially for metal components, using currently available X-ray tomography systems, X-ray path lengths limit the amount of material that can be penetrated with a given X-ray source. A full tomographic inspection requires the acquisition of X-ray images through an object from evenly and closely spaced directions measured along an orbit around the object, [8, 9, 10, 7, 11, 12]. For many objects, large aspect ratios or complex internal structures that lead to widely varying path lengths through the material, often prevent the acquisition of X-ray observations from certain directions. Furthermore, in a production line setting, inspection speed is often a key factor, which leads to a desire to reduce the number of X-ray measurements that are taken during a scan. In this paper, we are thus interested in the characterisation of the porosity distribution of a component if we are only able to collect limited X-ray measurements. We here distinguish and study two distinct cases of interest in different manufacturing settings. In the first setting, we have limited measurements, where the measurements are nevertheless evenly distributed in the orbit around the object. We call this setting the limited measurement setting. In the second setting, we are only able to collect data

from a limited range of projection angles. This setting will be called the limited angle setting. In both of these settings, using standard tomographic reconstruction methods leads to volumetric images with significant noise and artefacts, which complicate further analysis and thus prevents detailed porosity analysis. To overcome these issues, we explore the use of machine learning techniques, where we utilised known training data, for which we have both, a degraded volumetric image as well as a high quality "ground truth" image to train an algorithm to remove the noise and artefacts from the data.

If high quality tomographic data is available, then traditional image processing methods often provide good porosity estimates, though machine learning based approaches have recently been shown to offer some benefits [13]. We here extend this work in several ways. 1) Instead of working with 2D data slices, we work directly with a 3D volumetric data model; 2) we apply these methods to the much more challenging limited data settings outlined above and 3) we train the developmed method on a single volumetric image.

2. X-ray tomography

X-ray Cone Beam Computed Tomography (CBCT) [8] inspection can be performed using a fixed mini- or micro-focus X-ray source and fixed digital flat panel X-ray detector. The object under inspection is placed on a turntable between the X-ray source and the X-ray detector and X-ray projection measurements are taken through the object for equally spaced object rotation angles, typically covering a full 360 degree rotation. Individual projection images are then processed and combined using the Feldkamp, Davis, Kress (FDK) algorithm [14] to construct a 3D volumetric image of the spatial distribution of the object's X-ray absorption properties [8]. With this method, a rule of thumb states that in order to achieve a desired resolution R_s for an object with diameter d, we would require at least $\pi d/(2R_s)$ evenly spaced projections. With fewer projections, resolutions achieved with the FDK method decrease, whilst image noise increases. Furthermore, if it is not possible to collect X-ray projections for certain directions, then the image resolution in the direction orthogonal to the missing projection angle decreases [15].

2.1. Regularisation and artefact reduction

To deal with limited measurements, one approach is to formulate the image recovery problem as a regularised inverse problem that balances an

observation noise term and a regularisation penalty, which encodes prior knowledge about the image to be recovered. Here, total variation penalties, that encourage smooth images with sharp edges are commonly found [15]. However, solving the penalised inverse problem leads to iterative computational methods that are very slow and require significant computing resources. These methods are thus still seldom applied in realistic applications.

An alternative approach, that we will use here, is one based on image denoising. The relatively fast FDK method is used to estimate a noisy image with artefacts and a noise and artefact removal algorithm is then used in a separate step.

2.2. Machine learning for XCT

Machine learning methods are now routinely applied to many tomographic inverse problems. The most advanced methods to date typically build deep networks that are unrolled versions of classical optimisation strategies, but where training data is used to optimise the network, which effectively offers learned regularisation to enforce the structures found in typical training data [16]. An alternative approach is to directly train an inverse mapping, that takes the tomographic measurements and predicts an image. Whilst these method can work well on 2D images or very small 3D images, it is currently not possible to apply these methods to realistically sized 3D data that is found in typical XCT based inspection settings. Furthermore, most of these methods rely on significant training data, which can be difficult to collect in real applications. Machine learning based methods for XCT image reconstruction of realistically sized data thus currently rely on de-noising approaches and these methods have gained increased prominence recently, where modern, machine learning based artefact removal methods have been found to be particularly powerful. For example Pelt et al. [17] have trained a mixed scale network architecture. Whilst the model requires reduced computing memory, it has so far only been applied to 2D slices taken from a 3D reconstruction. An alternative or complementary set of approaches works in the observation domain. For example, a learned de-noising methods could be applied to reduce the noise from low-dose projection images before the FDK reconstruction as in [18], or we could use a trained model to optimally interpolate observations to fill in missing data [19].

3. Models

We here instead follow a slightly different image de-noising approach, where we train a de-noising algorithm to work directly on 3D image data. It is hoped that such an approach can learn more complex 3D image structures, which might then lead to more accurate de-noising results. However, we also want to 1) limit the required training data and 2) keep the model size small, so that it can be implemented with standard computing hardware. We thus opt for a model that operates only on smaller 3D image blocks. Let $B_b(I, i, j, k) = I[i : i + b, j + b, k + b]$, be the operator that cuts out a $b \times b \times b$ block of image $I \in \mathbb{R}^{N_x \times N_y \times N_z}$ with block corners at pixels (i, j, k) and $(i + d, j + d, k + d)$. We then train different 3D image models on the training set $\{B_{32}(I, i, j, k)\}_{i \in \mathcal{I}, j \in \mathcal{J}, k \in \mathcal{K}}$, where $\mathcal{I} = \{0, b/s, 2b/s, N_x - b\}, \mathcal{J} = \{0, b/s, 2b/s, N_y - b\}$ and $\mathcal{J} = \{0, b/s, 2b/s, N_z - b\}$ and where s is the block overlap set to 2 for training. Such a block based approach is similar to the 2D slice based approach of [17] in that an initial FDK reconstruction is de-composed into smaller subsets, with each subset of the data then being processed independently. However, whilst for 2D slice based processing, each image subset is independent, for our block based approach, overlapping blocks can be extracted from the image. For inference of the entire image, we use the trained model and apply it to all image blocks extracted with a given overlap fraction s, apply the model to each block and then estimate the de-noised image by averaging the corresponding voxel over all individual image blocks that contain this voxel. Whilst increasing the computational demand (each pixel is de-noised several times), this reduces visible image artefacts at block boundaries and also has the potential to further reduce noise, as voxel estimates are now derived from several different estimates.

We implemented and compared several deep models, a 3D U-net based model, that uses an architecture similar to [20], a 3D convolutional autoencoder [21] as well as 2D and 3D versions of the method in [17].

3.1. U-net architecture

We implemented a standard U-net architecture [20], but using 3D convolutional layers instead of the more standard 2D convolutions. In particular, the U-net is constructed of 3D convolutional layers, each using a 3×3×3 kernel size and zero padding. A convolution block is made up of two blocks

each consisting of a convolution layer, batch normalisation and a ReLU non-linearity. We use 4 encoder blocks, where each block uses the above convolution block followed by a 2×2×2 max pooling block. The first block has 8 output channels, with each subsequent block doubling the number of output channels so that the fourth block has 64 channels, which are then fed into a convolution block with 128 output channels.

There are also 4 decoding blocks, concatenating the output of the upsampled previous block and the output of the corresponding encoder block before max pooling. The concatenated channels are send through a convolution block, with output channels being half the number of input channels. The last decoder block has 8 output channels, which are send through a single convolution layer with kernel size 1, followed by a final ReLU nonlinearity to guarantee positive outputs.

We also trained a 2D U-net, following the structure in [13], but instead of training the network as a classifier (i.e. with a sigmoid non-linearity in the output and using the binary cross-entropy loss), we instead trained the model as an image de-noiser in the same way as our other networks, using a ReLU non-linearity in the output and the mean squared error loss function.

3.2. Autoencoder Architecture

The autoencoder is build out of the same encoder structure, using the same 3D convolution architecture as the U-net, with the only difference being that we do use strided 3D convolutions (with a stride of 2) in the second convolution of each convolution block instead of the max pooling layers. This generates 64 channels in the code. The decoder uses the inverse structure of the encoder, using transposed convolutions. The autoencoder also does not use any skip connections.

3.3. Mixed-Scale Dense Convolutional Neural Network (MSDC-net)

For comparison, we also implement the 2D convolutional Mixed-Scale Dense Convolutional Neural Network (MSDC-net) of [17] using 100 layers as in [17] . Even when using a batch size of 1, we were unable to train this model on the GPU we had available when trying to learn a model for a full 2D slice of our data-sets. We thus also used a block wise training strategy, but using blocks from the 2D image slices of size 512×512. This model had significantly more parameters than our U-net and Autoencoder models and training was thus much slower. To more deircetly compare this architecture to the 3D Unet and autoencoder, we also implemented the same

model using 3D convolutional layers trained on 3D blocks of size 32×32×32, where training this time was done using a batch size of 8.

4. Data sets

We use two different data-sets to evaluate our method. The first data-set is simulated, allowing us to generate data with uniform distributions of pore-like features, with the ability to vary pore size and density as well as image noise. The second data-set is real data as described in [13].

4.1. Simulated data

To generate simulated data that allows us to simulate porosity similar to that observed in [6], we proceeded as follows:

1. We generated a volumetric dataset of size 2000×2000×2000 where each value is drawn from an independent zero mean, unit variance Gaussian distribution.
2. The data is then filtered with a 3D convolution using a Gaussian kernel with standard deviation σ. Increasing σ generates larger features.
3. The data is then thresholded, with values above the threshold set to 1 and values below the threshold set to 0. Note, a threshold of 0 will mean that 50% of the data is 0 and 50% is 1, whilst, due to the Gaussian nature of the data, a threshold of 1 (i.e. one standard deviation) leads to 84.1% of values set to 1.
4. To simulate a cylindrical object, we take each x-y slices of the data and mask out the pixels that are further than 1000 pixels away from the centre of each slice.

To simulate realistic tomographic images, we generate projection images from the volumetric data using the TIGRE tomographic imaging toolbox [22]. We set up a scan geometry where the volumetric image has a side-length of 10mm, We set the source to object distance to 10mm and the source to detector distance to 250mm. the detector simulates a flat panel detector with 2000×2000 pixels, each of size 0.2mm by 0.2mm.

We generate two different scan scenarios, a limited number of projection scan, where we simulated 128 projections, equally spaced between 0 and 360 degrees around the object. We also simulated a reduced angle scan, where we collected 785 equally spaced projections that only covered 90 degrees. For both of these data-sets, we added poisson random noise to the projections by

setting the minimum X-ray transmission through the object to 10% of the maximum transmission. We then assumed that the maximum transmission had a mean X-ray flux of λ X-ray photons. By varying λ different amounts of noise could be added. We here report results computed with λ=60000 unless stated otherwise. The projection data was then reconstructed using the FDK algorithm [14] using the implementation available in [22].

4.2. Real data

We also had access to the reconstructed volumetric data of [13], which is a data-set with relatively little noise and very few artefacts. To simulate limited angle and limited measurement data, we applied the same process to this data that was used for the simulated data, namely, we computed projection images simulating the scanning system used in [13] by using the TIGRE toolbox with a detector size of 1000×1000 pixels, each of 0.1mm side length, a 3D volume size of 15mm×15mm×15mm, a source to object distance of 34mm and a source to detector distance of 154mm. We also added poisson noise to these projections (using $\lambda = 60000$) and then used either 128 equally spaced projections over a 360 degree range or 785 projections over a 90 degree range. Reconstructions were again computed using the FDK method.

4.3. Training

All models were trained on a single volumetric image. We extracted all 32×32×32 sized block with an overlap of 50% (unless stated otherwise) and split the data into a 90% training set and a 10% validation set to monitor training. For the original 2D version of MSDC-net we used blocks of size 512×512 as described above. All models were trained using the ADAM optimiser with a learning rate of 0.0001 and weight decay of 0.00001 and a batch size of 64, apart from the MSDC-net, which used a batch size of 1 for the 2D version and a batch size of 8 for the 3D version. We trained each model for 50 epochs. Training was performed on a linux workstation using 24 AMD EPYC 7401P CPU cores, 256 GB of RAM and two GeForce RTX 2080 GPUs with 12GB of memory each (only one of which was used during training). Code was implemented in python using pytorch to define, train and test the different machine learning models.

4.4. Data normalisation

When training networks on a single volumetric image using a block-based approach, image normalisation cannot be done on a block by block basis,

instead, we here use a global image normalisation approach. Whilst global 3D image normalisation using a min-max scaler worked well on the simulated data, it is less robust to outliers and so, we found dividing the image intensity by the image standard deviation worked well and is thus used for the real data.

5. Results

5.1. Artificial data

We start by evaluating the method on artificial data, where the ground truth is known and where we have control over pore size and density.

5.1.1. Comparison of networks and influence of variations in the pore size and density

We trained three different models on data generated with a porosity of 50% and with medium size pore sizes ($\sigma = 5$). In particular, we trained the 3D U-net, the 3D Autoencoder and the 2D U-net of [13]. We here used the data reconstructed from 128 projections using the FDK method. We also compare the results to the use of global thresholding, where we determined the threshold using Otsu's method [23]. Results, evaluated on new data over a range of different porosity statistics are shown in tables 1, 2, 3 and 4 respectively, where performance is evaluated in terms of classification accuracy. The 3D U-net outperforms all other methods, apart from the very low porosity data, where the 2D U-net shows a very small improvement. Given that the U-nets outperform the auto-encoder model and in many settings performs noticeably better than the 2D model, we concentrate on the 3D U-net model for most of the rest of this paper. Note also, that none of the learned models performed well when estimating data with many pores that were smaller than those used in training, a case were even simple thresholding outperformed the machine learning based methods. As we here trained the model on larger pores, it seems that the model learned a level of image smoothness and thus seems to average out smaller image features. Note that, results for direct thresholding of the FDK method can further be improved if we use an additional median filtering step (results not shown here), though the machine learning methods still preform better and thus seem to offer additional de-noising benefits compared to standard filtering.

Results achieved with the 3D U-net de-noising approach and direct thresholding can visually be compared in figure 1, where we show results for a single

	$\sigma = 10$	$\sigma = 5$	$\sigma = 1$
0.1% porosity	0.9992	0.9989	0.9982
13.6% porosity	**0.9869**	**0.9631**	**0.8629**
50% porosity	**0.9573**	**0.8835**	0.5946

Table 1: Classification accuracy achieved with the U-net trained as a de-noiser followed by global thresholding for classification. Numbers in bold indicate cases where the method outperformed the other de-noising methods.

	$\sigma = 10$	$\sigma = 5$	$\sigma = 1$
0.1% porosity	**0.9993**	**0.9990**	**0.9984**
13.6% porosity	0.9788	0.9475	0.7579
50% porosity	0.9426	0.8676	0.5552

Table 2: Classification accuracy achieved with the 2D U-net trained as a de-noiser followed by global thresholding for classification. Numbers in bold indicate cases where the method outperformed the other de-noising methods.

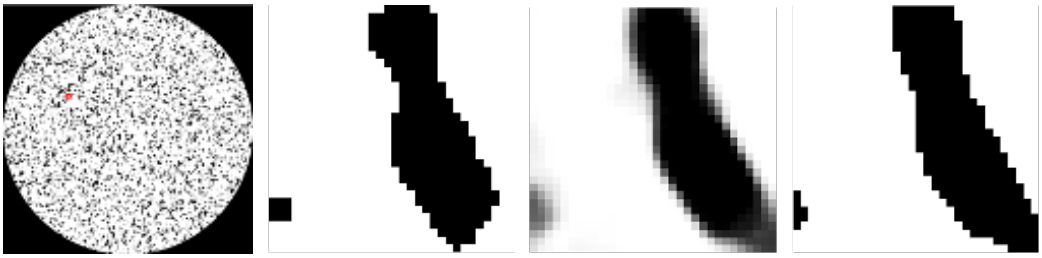
	$\sigma = 10$	$\sigma = 5$	$\sigma = 1$
0.1% porosity	0.9991	0.9988	0.9981
13.6% porosity	0.9843	0.9589	0.7737
50% porosity	0.9544	0.8771	0.5942

Table 3: Classification accuracy achieved with the auto encoder trained as a de-noiser followed by global thresholding for classification. Numbers in bold indicate cases where the method outperformed the other de-noising methods.

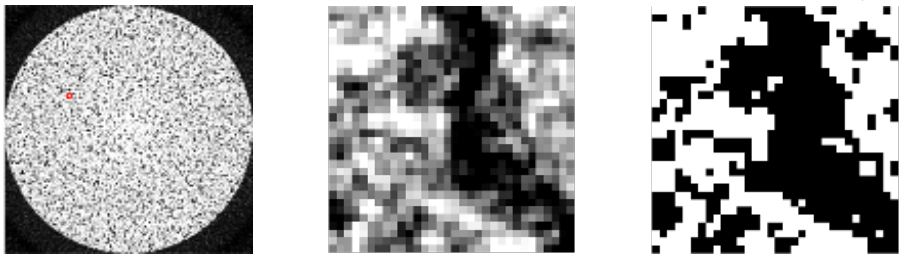
	$\sigma = 10$	$\sigma = 5$	$\sigma = 1$
0.1% porosity	0.5617	0.5619	0.5605
13.6% porosity	0.6641	0.6452	0.5665
50% porosity	0.7847	0.7439	**0.6286**

Table 4: Classification accuracy achieved with global Otsu thresholding. Numbers in bold indicate cases where the method outperformed the other de-noising methods.

slice through one block of the data and figure 2, where we show results for an entire slice through the volume, this time, for data where pore sizes are much larger than those used in training. A more detailed analysis of the threshold choice in the methods can be seen by looking at a full ROC curve evaluated on a randomly chosen data block as shown in figure 3, which is the result



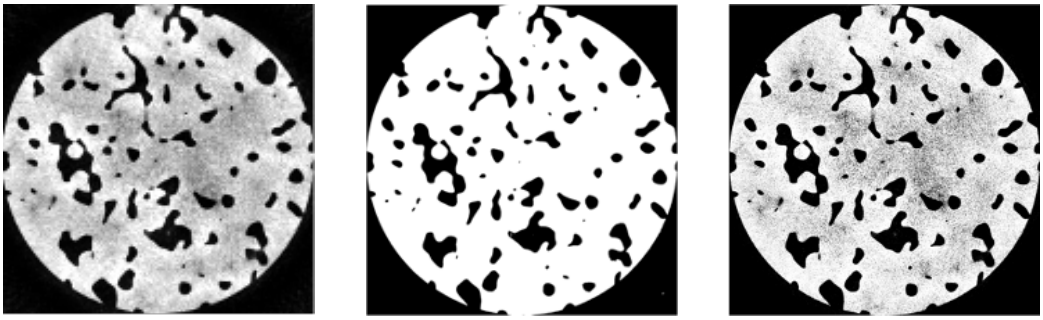
(a) Original (full slice) (b) Original (block) (c) Our method (block) (d) Our method + thresholding (block)



(e) FDK reconstruction (full slice) (f) FDK reconstruction (block) (g) Otsu Thresholding of FDK

Figure 1: Slices through the volume comparing our U-net based estimate to thresholding results where the threshold is computed using Otsu's method. Here the test data has similar pore statistics as the training data. Shown are 2D slices from the volumetric data. Red squares in left column indicate location of the blocks shown in the remaining panels.

where the pore size is the same between training and test data.



(a) FDK reconstruction (full slice) (b) Our method + thresholding (full slice) (c) Global Otsu thresholding (full slice)

Figure 2: Slices through the volume comparing our U-net based estimate to thresholding results where the threshold is computed using Otsu's method. Here the test data has larger pores than the training data. Shown are 2D slices from the volumetric data.

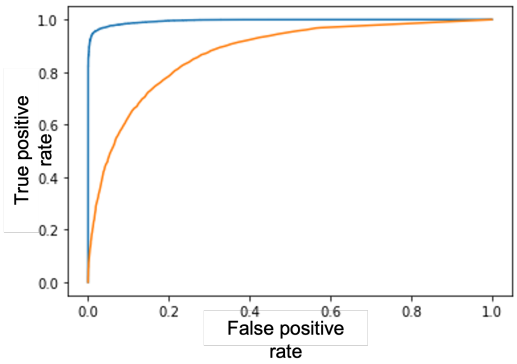
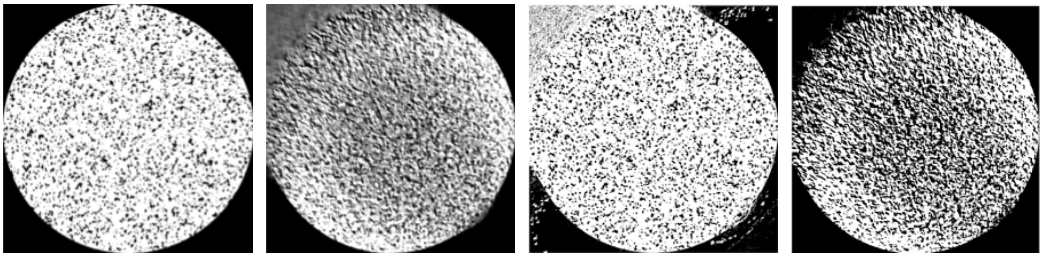


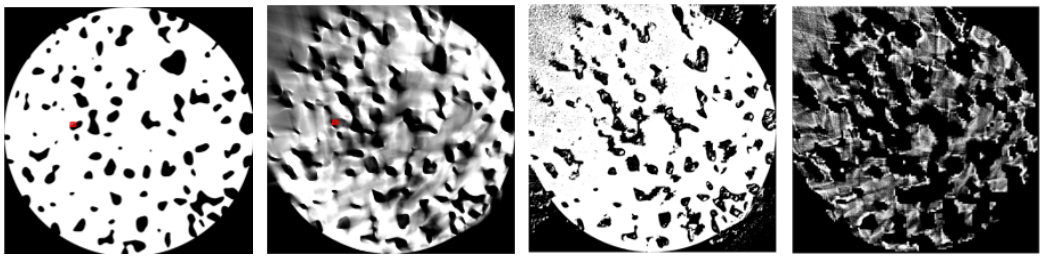
Figure 3: ROC curve evaluated on a single block comparing our method (blue) to direct thresholding (orange).



(a) Original (full slice) (b) FDK reconstruction (full slice) (c) Our method + thresholding (full slice) (d) Global Otsu thresholding (full slice)

Figure 4: Slices through the volume comparing our U-net based estimate to thresholding results where the threshold is computed using Otsu's method. Here the test data is from a limited angle scan, where the poe statistics are the same between the training data and the test data.

Looking at the limited angle dataset, where projections are only available over a 90 degree range, similar benefits of the 3D block based U-net model can be observed. We show full image slices from the 3D data in figures 4 and 5, showing the original data, the FDK reconstruction from limited angle data as well as the 3D U-net based results as well direct thresholding of the FDK reconstruction. Results here are shown for a dataset that has the same pore size and larger pore sizes compared with the training data respectively. The numerical performance, independent of the threshold choice, is shown in the ROC curves in figures 6 and 7, which again clearly demonstrates the benefits of using the block based 3D U-net over direct thresholding.



(a) Original (full slice) (b) FDK reconstruction (full slice) (c) Our method + thresholding (full slice) (d) Global Otsu thresholding (full slice)

Figure 5: Slices through the volume comparing our U-net based estimate to thresholding results where the threshold is computed using Otsu's method. Here the test data is from a limited angle scan, where the pores are larger than those used in model training.

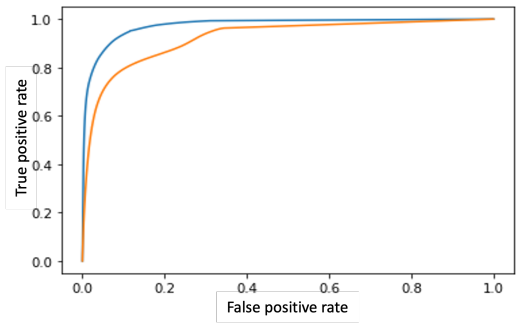


Figure 6: ROC curve evaluated on a full slice comparing our method (blue) to direct thresholding (orange) on the limited angle scan data with similar pore size.

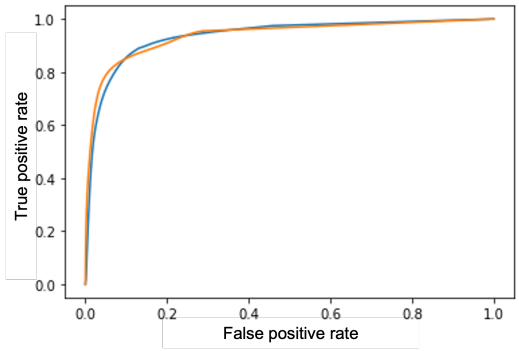


Figure 7: ROC curve evaluated on a full slice comparing our method (blue) to direct thresholding (orange) on the limited angle scan data with larger pore size.

5.1.2. Classification vs de-noising

In [13], the proposed 2D network was trained as a classifier rather than a de-noising algorithm. To evaluate potential benefits of each approach, we trained the 3D U-net also as a classifier as well as a de-noising model (changing the output non-linearity as well as cost function). Again, the data for training was generated using a threshold of 0, (i.e. 50% porosity) and medium size pore sizes ($\sigma = 5$). We then evaluated the performance of both models on new data over a range of porosity fractions and pore sizes. Results are shown in tables 5 and 6, where we measure performance using the classification accuracy. Whilst there are small differences in performance between these two models, these differences are small compared to the variation observed between conditions. Interestingly, comparing the results where the de-noising based U-net performed well to the results found with the other de-noising methods, we see that the de-noising Unet works better than the classification U-net exactly for those case where the de-noising U-net worked less well than the other de-nosing methods methods.

	$\sigma = 10$	$\sigma = 5$	$\sigma = 1$
0.1% porosity	**0.9992**	**0.9989**	**0.9982**
13.6% porosity	0.9869	0.9631	**0.8629**
50% porosity	0.9573	0.8835	**0.5946**

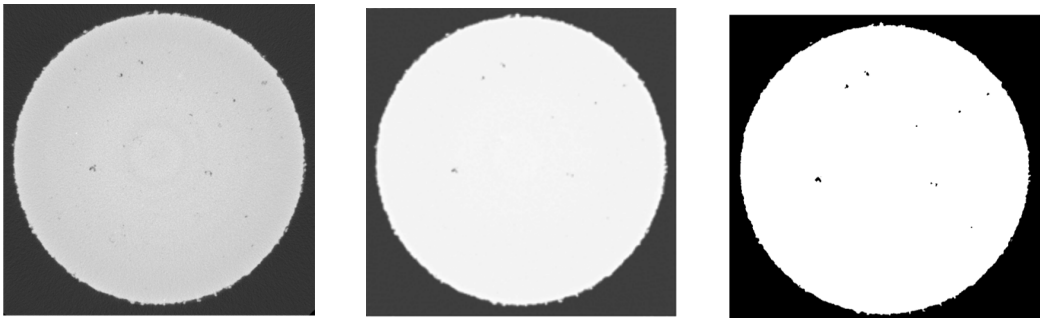
Table 5: Results of classification accuracy achieved with the U-net trained as a de-noiser followed by global thresholding for classification. Numbers in bold indicate cases where the training method outperformed the classification based training.

	$\sigma = 10$	$\sigma = 5$	$\sigma = 1$
0.1% porosity	0.9935	0.9935	0.9920
13.6% porosity	**0.9873**	**0.9645**	0.8460
50% porosity	**0.9599**	**0.8978**	0.5942

Table 6: Results of classification accuracy achieved with the U-net trained as a classifier followed by global thresholding for classification. Numbers in bold indicate cases where the training method outperformed the de-noising based training.

5.2. Real data

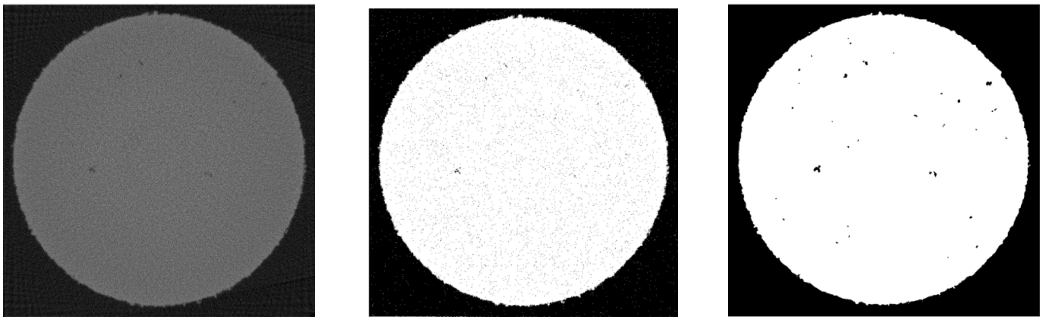
Pore structures in the real data are likely to be somewhat different from the Gaussian random field structure used to generate our training data.



(a) Original high quality reconstruction (full slice)

(b) Our method (full slice)

(c) Our method + thresholding (full slice)



(d) FDK reconstruction (full slice)

(e) Otsu's Thresholding (full slice)

(f) Hand segmentation of high quality data (full slice)

Figure 8: Slices through the volume comparing our U-net based estimate to thresholding results where the threshold is computed using Otsu's method. Here the test data is generated from real scan data from [13] by re-projecting and FDK reconstructing with 128 projections. Shown are 2D slices from the volumetric data.

5.2.1. Application to limited numbers of projection real data

We again start by looking at the simpler case where we have only 128 projections collected over 360 degrees. A full 2D slice of the data is shown in figure 8, where we show the original high quality data, the FDK reconstruction from limited measurements, the de-noised FDK reconstruction using our method, a thresholded version of our de-noised method as well as the results obtained by using global thresholding. Results are compared visually to a hand segmentation of the volume as provided by [13].

The ROC curves for these two methods, as well as global thresholding applied after the use of a 2D median filter with a 11×11 square kernel are shown in figure 9, where it is clear that the machine learning based approach

has significant benefits compared to global thresholding, though for the data reconstructed from limited projections here, there is only a small benefit in using the machine learning based de-noiser over the simpler median filter.

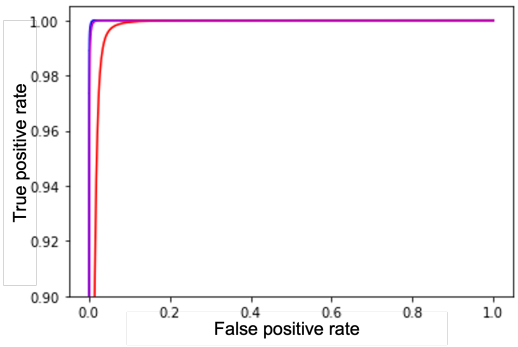
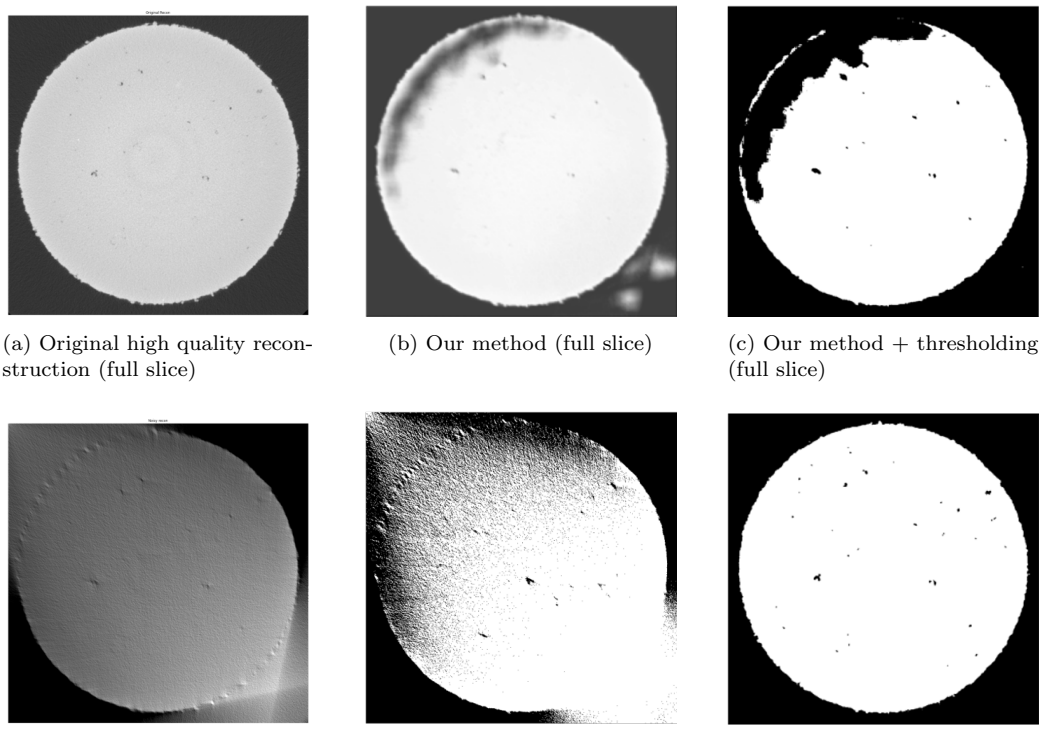


Figure 9: ROC curve evaluated on a full slice comparing our method (blue) to direct thresholding (orange) on the scan data generated from a real scan using only 128 projections. Also shown is a thresholding estimate applied after median filtering of the image using an 11×11 square kernel (magenta).

5.2.2. Application to limited angle real data

Results look however much more beneficial for the machine learning based approach when looking at data with significant limited angle artefacts as shown in figure 10, where we see visually, that the block based machine learning based approach is able to identify some areas that are outside the object, but where limited angle artefacts lead to FDK reconstructions with voxel values similar to those seen inside the object and so could not be distinguished based on thresholding alone. Whilst the method identifies some of the internal object as being a void, the results are significantly better than could have been achieved with thresholding. As our machine learning method analyses individual blocks, it seems that the method learns the fine structure of the anisotropic noise variations, both inside and outside the object in order to distinguish which blocks are inside and which are outside, as this distinction cannot be made based on overall grey level values alone (the top left corner of the FDK image has the same average grey value as the inside of the object).

This is also clear by looking at the ROC curves (see figure 11), where again, thresholding (even with median filtering) can simply not distinguish



(a) Original high quality reconstruction (full slice)

(b) Our method (full slice)

(c) Our method + thresholding (full slice)

(d) FDK reconstruction (full slice)

(e) Otsu's Thresholding (full slice)

(f) Hand segmentation of high quality data (full slice)

Figure 10: Slices through the volume comparing our U-net based estimate to thresholding results where the threshold is computed using Otsu's method. Here the test data is generated from real scan data from [13] by re-projecting and FDK reconstructing from a limited angle scan spanning only 90 degrees. Shown are 2D slices from the volumetric data.

the object's outside from the inside, whilst the machine learning method preforms significantly better.

5.2.3. Comparision to 2D MSDC-net

We also trained the MSDC network as described above, both on the limited angle (90 degrees) data as well as the reduced number of projection data (128 projection). ROC curves for a slice cleaned up using the MSDC-net followed by thresholding is shown in figure 12 and 13 for the 128 projection and the 90 degree data respectively. Again, for the data generated from 128 projections, the machine learning methods perform very well and there is little difference between the results for the 3D block based U-net results and

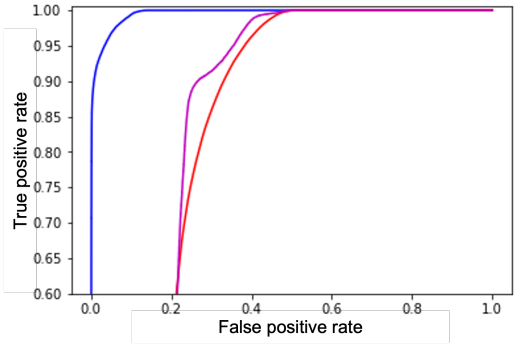


Figure 11: ROC curve evaluated on a full slice comparing our method (blue) to direct thresholding (orange) on the limited angle scan data generated from a real scan. Also shown is a thresholding estimate applied after median filtering of the image using an 11×11 square kernel (magenta).

the 2D MSDC-net. The advantages of the 3D block based U-net de-noising is however much more evident for the significantly more challanging limited angle data, where the MSDC-net performed poorly compared to the 3D block based U-net data.

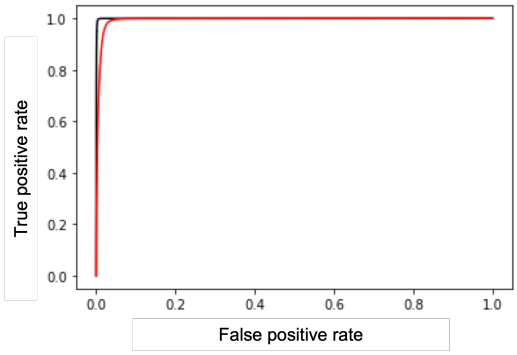


Figure 12: ROC curve evaluated on a full slice comparing the MSDC-net (black) to direct thresholding (red) on the scan data generated from a real scan using only 128 projections. Not shown here are the results of our method, but the ROC curve here is visually indistinguishable from the MSDC-net result shown and is thus omitted.

5.2.4. Comparison to 3D MSDC-net

As a final comparison we also trained a 3D block version of the MSCD net, where we replaced the 2D convolutional layers in the network with 3D

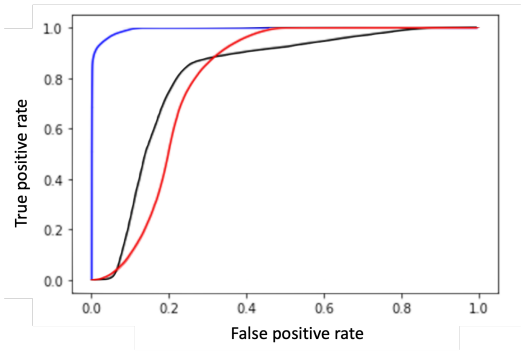


Figure 13: ROC curve evaluated on a full slice comparing our method (blue) to the 2D MSDC-net (black) and direct thresholding (red) on the limited angle scan data generated from a real scan. A

layers. We then trained on 32 $\times$32$\times$32 voxel blocks in the same way as with the 3D U-net and Autoencoder models. We here used the same real data reconstructed from projections limited to 90 degrees as above, following the exact same approach for training and testing, just with the different model. Due to the high number of parameters, training took significantly longer (just over 8 days for 50 epochs). A reconstructed slice is shown in figure 14, which should be compared to the same slice as reconstructed with the U-net shown in Figure 10b. To evaluate the performance numerically, we again show the ROC curves, comparing the 3D MSDC net, the 3D Unet and direct threshiolding in figure 15

6. Conclusions

Whilst X-ray tomographic image reconstruction has been shown to benefit from data-driven methods, applying these to realistically sized data remains challenging, whilst training still requires significant amounts of high quality data that is not normally available. In this paper we have shown how a learned de-noising approach can be trained on a single volumetric image and then applied to a full 3D tomographic reconstruction. Working with smaller 3D blocks was shown to allow us to train the method on as single image whilst at the same time providing substantial benefits over alternative approaches in terms of the detection of internal object defects.

There remain however several challenges, especially in limited angle scan settings, where significant data is missing. Whilst our results have shown

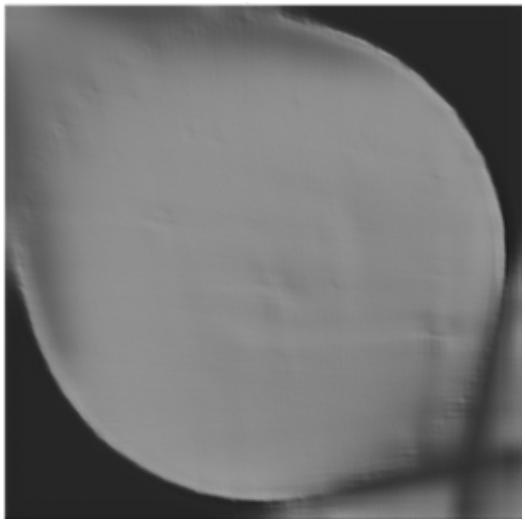


Figure 14: Slice from the 3D volume reconstructed with the FDK method and de-noised using a 3D MSCD network with 100 layers applied to overlapping image blocks of size 32×32×32.

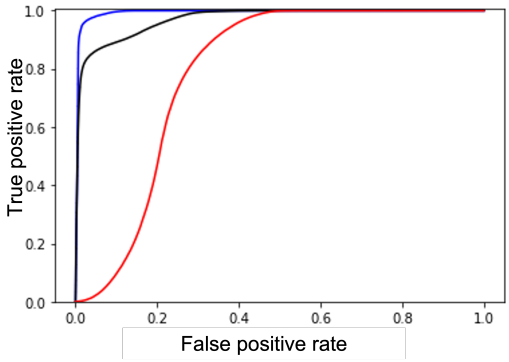


Figure 15: ROC curve evaluated on a full slice comparing our method (blue) to the 3D MSDC-net (black) and direct thresholding (red) on the limited angle scan data generated from a real scan. A

strong benefits here, these heavily relied on the availability of training data where high quality data was available. Whilst we have reduced training data requirements to a single image, to apply these methods in real applications, it will be crucial to collect this training data image in a similar setting to the real data, which might still remain difficult in certain applications.

We have here used a local block model, that works on blocks with 32 pixel

side length. This model is obviously only able to capture local structure on this scale (e.g. local noise properties, local X-ray attenuation properties and material boundaries), so won't be able to capture larger features. In applications were larger features exist, we could either use larger blocks or use a second model that is applied to larger blocks. To keep the computational benefits of the small models used here, such an additional model could be applied to a downsampled version of the image, where we first model the large, low frequency features on a larger scale and then fill in the variation on a smaller scale. Using two independent model, whilst being more efficient, would not be able to capture statistical dependancies between these features, though for the manufactured object inspection application studied here, local material density variation and material fine structure might not be correlated strongly to overall object shape properties. The problem with such an approach might however be that we would not be able to train on a single image, as we have far fewer blocks at this scale in any one image.

Acknowledgment

This research was funded by the UK Research Centre in Non-destructive Evaluation (RCNDE) and EPSRC grant EP/R002495/1. The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of a Titan Xp GPU used for this research.

References

[1] W. Frazier, Metal additive manufacturing: a review, J. Mater. Eng. 23 (2014) 1917–1928.

[2] I. Gibson, D. Rosen, B. Stucker, Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing, Springer, New York, 2014.

[3] W. Sames, F. List, S. Pannala, R. Dehoff, S. Babu, The metallurgy and processing science of metal additive manufacturing, Int. Mater. Rev. 62 (2016) 315–360.

[4] T. Debroy, H. Wei, J. Zuback, T. Mukherjee, J. Elmer, J. Milewski, A. Beese, A. Wilson-Heid, A. De, W. Zhang, Additive manufacturing of metallic components – process, structure and properties, Prog. Mater. Sci 92 (2017) 112–224.

[5] B. Zhang, Y. Li, Q. Bai, Defect formation mechanisms in selective laser melting: a review, Chin. J. Mech. Eng. 30 (2017) 515–527.

[6] F. Kim, M. S.P., E. Garboczi, J. Slotwinski, Investigation of pore structure in cobalt chrome additively manufactured parts using x-ray computed tomography and three-dimensional image analysis, Additive Manufacturing 17 (2017) 23–38.

[7] A. Thompson, I. Masker, R. Leach, X-ray computed tomography for additive manufacturing: a review, Meas. Sci. Technol. 27.

[8] A. C. Kak, M. Slaney, Principles of Computerized Tomographic Imaging, IEEE Press, 1988.

[9] L. De Chiffre, S. Carmignato, J.-P. Kruth, R. Schmitt, W. A., Industrial applications of computed tomography, CIRP Ann. Manuf. Technol. 63 (2014) 655–677.

[10] W. Wits, S. Carmignato, F. Zanini, T. Vaneker, Porosity testing methods for the quality assessment of selective laser melted parts, CIRP Ann. Manuf. Technol. 65 (2016) 201–204.

[11] A. Du Plessis, I. Yadroitsev, I. Yadroitsava, L. R. S.G., X-ray microcomputed tomography in additive manufacturing: a review of the current technology and applications 3d print, Addit. Manuf. 5 (2018) 227–247.

[12] A. Du Plessis, Standard method for microct-based additive manufacturing quality control 1: porosity analysis, MethodsX.

[13] C. Gobert, A. Kudzal, J. Sietins, C. Mock, J. Sun, B. McWilliams, Porosity segmentation in x-ray computed tomography scans of metal additively manufactured specimens with machine learning, Additive Manufacturing 36.

[14] L. Feldkamp, L. C. Davis, J. W. Kress, Practical cone-beam algorithm, Journal of The Optical Society of America A-optics Image Science and Vision 1 (1984) 612–619.

[15] F. Natterer, The Mathematics of Computed Tomography, SIAM, 2001.

[16] S. Arridge, P. Maass, O. Öktem, C. Schönlieb, Solving inverse problems using data-driven models, Acta Numerica 28.

[17] D. M. Pelt, K. J. Batenburg, J. A. Sethian, Improving tomographic reconstruction from limited data using mixed-scale dense convolutional neural networks, JOURNAL OF IMAGING 4 (11). doi:10.3390/jimaging4110128.

[18] X. Yang, V. De Andrade, W. Scullin, E. L. Dyer, N. Kasthuri, F. De Carlo, D. Gursoy, Low-dose x-ray tomography through a deep convolutional neural network, Scientific Reports 8. doi:10.1038/s41598-018-19426-7.

[19] E. Valat, K. Farrahi, T. Blumensath, Data-driven interpolation for super-scarce x-ray computed tomography (2022). doi:10.48550/ARXIV.2205.07888.
URL `https://arxiv.org/abs/2205.07888`

[20] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, MICCAI 2015, 2015, pp. 234–241.

[21] V. Jain, H. S. Seung, Natural image denoising with convolutional networks, in: Proceedings of the 21st International Conference on Neural Information Processing Systems, NIPS08, 2008, pp. 769–776.

[22] A. Biguri, R. Lindroos, R. Bryll, H. Towsyfyan, H. Deyhle, I. Harrane, R. Boardman, M. Mavrogordato, M. Dosanjh, S. Hancock, T. Blumensath, Arbitrarily large tomography with iterative algorithms on multiple gpus using the tigre toolbox, J. Parallel Distributed Comput. 146 (2020) 52–63.

[23] N. Otsu, A threshold selection method from gray-level histogram, IEEE Trans. Syst. Man Cybern. 9 (1979) 62–66.