Internal Bone Reconstruction Using Machine Learning

Moira Shooter[*]
The National Centre for Computer Animation
Bournemouth University, UK
Poole, Dorset
moira.shooter@hotmail.com

Lucien Hugueniot[†]
The National Centre for Computer Animation
Bournemouth University, UK
Poole, Dorset
lucien98@gmail.com

ABSTRACT

The goal of our research is to determine whether machine learning can predict the porosity factor to 3D construct the internal structure of a porous object such as a bone. This is valuable as there are many fields that require porous models to be accurately represented such as in computer graphics, to simulate objects in a more precise way; to reconstruct synthetic soils to optimise plant growth; in the medical area to 3D print prosthetics to cure bone tumors. There is a high necessity of large datasets for machine learning algorithms to work. Therefore our first step was to find a large enough data set. This consisted of ct-scans of sick, healthy and transplanted mice tibia bone cross-sections , then we processed the images in order to calculate the porosity factor and labelled each image depending on the porosity factor and health factor. To expand our dataset we modified the images by flipping, translating, rotating them and varied the brightness on them. The next step was to train the networks, one that would recognise the porosity of a bone and another that would recognise the healthiness of a bone. We tweaked the neural network values until we were satisfied with the recognition accuracy. Finally we show how good he machine learning algorithm has predicted successfully the porosity factor of an object and its healthiness.

CCS CONCEPTS

• **Computing methodologies → Machine Learning**.

KEYWORDS

Machine learning, 3D reconstruction, porous material

ACM Reference Format:
Moira Shooter and Lucien Hugueniot. 2019. Internal Bone Reconstruction Using Machine Learning.

1 INTRODUCTION

This paper describes an original approach to the 3D reconstruction of porous objects, our method involves using machine learning to aide in the reconstruction process. The realm of 3D reconstruction using machine learning has been increasing ever since hardware has made it a practical solution to problems of this field. Open computer vision software is constantly being updated, this allows anyone to use cutting edge technology at a relatively affordable price. The automotive industry is using machine learning and 3D construction to avoid car collisions, many scientific fields require a lot of manual work for the analysis of 2D data that can in theory be automated. We have decided to take on this task in an attempt to push the boundaries of collaboration between man and machine.

The traditional approach to 3D reconstruction is to construct an object from sensor data, this could be optical, acoustic, laser scanning, radar, thermal or even seismic data. The data is then used to synthesise a 3D object using techniques such as photogrammetry. Often these objects are inconsistent and do not reflect the real life object they are trying to imitate. Machine learning has a propensity for handling large, inconsistent sets of data. Convolutional neural networks have proved to be especially useful for the processing of organic object data .
We propose to answer the following question: Is it advantageous to use Machine Learning for the reconstruction of 3D volumetric objects? To answer this we have decided to use organic bones as a case study. The medical field could be a great beneficiary of accurate 3D bone reconstruction, for example a patient might need Osseointegration[7] implant surgery. In practice, the implant's structure is artificial and does not adhere to the rest of the bone's structure and can only be considered stable if it passes the "percussion analysis test", but by constructing the artificial piece using machine learning the implant has the potential to be a better fit for the bone.
Our contribution were to:

- Image process the data in a way to find the porosity factor
- Augment the data to enlarge the dataset and to have a higher accuracy from the neural networks
- Creating two neural networks based on what they predict, health and porosity factor, that we then use in a script that runs the two models, to predict the health factor of the bone and the porosity factor
- 3D reconstruct a bone based on the porosity factor and health factor

The paper is divided in 4 sections where we review the related work in Section 2. We quickly go over some machine learning technical terms in Section 3. Then we give a detailed overview of our approach in Section 4. Finally, Section 5 presents the results of our studies. Our code and pre-trained models are available at https://github.com/mshooter/MachineLearningBone.

2 RELATED WORK

Convolutional Neural Networks: Integration of machine learning and computer graphics attracted a lot of attention in the last

[*]Student
[†]Student

few years. To classify our images based on their health and porosity factor we use a type of neural network called convolutional neural network(CNN), CNNs are used for their quality, speed and simplicity. The first year that a convolutional neural network had outstanding predictions was in 2012 [5]. In the paper they discussed the architecture of the network which was called AlexNet. It was a relatively simple layout compared to todays architectures, the network was made up of 5 convolutional layers, max-pooling layers, dropout layers and 3 fully connected layers. The network was designed for classifcation with 1000 possible categories. The performance of convolutional neural networks doesn't always depend on the architecture of the network but it can also depend on the input images[2]. The main idea is that the approach transforms the input image in a way so that the subsequent layers have an easier time making a classification. We augmented our dataset for the purpose to get a better accuracy from the neural networks. We use 2D ct-scans of bone slices to feed our neural networks to predict the porosity or health factor of a bone slice. We chose this dataset because a bone is a porous object and our network needed to predict the porosity factor. Choosing a certain dataset depends on what you want the neural network to learn. Some other works depended on 3D inputs to reconstruct indoor scenes[16] or depended on one 2D density heat-map[17] to reconstruct 3D models instead of having multi-view images.

3D Reconstruction using machine learning: Reconstruction is a well-searched area that involves reconstruction of buildings, bones and other objects. One example of why we want to reconstruct objects such as a bone, is to reconstruct the skeleton of an animal or to have a more detailed view of the object to analyse it from different angles. Researches at the Technical University of Munich, Ludwig Maximilian University of Munich and Johns Hopkins University have produced a volumetric convolutional neural network called V-net[10] that performs native 3D segmentation of prostate MRI data. In 3D segmentation, the organ of interest is outlined and each of the voxels of the organ in the 3D image is grouped and assigned the same label. The V-net was trained on MRI volumes and learned to predict segmentation for the whole volume at once. This required a lot of computation therefore the researches used NVIDIA GPUs. Others[12] have tried to create a convolutional neural network that can be used to predict how an object from a 2D image would look in 3D. Their solution was to use two CNNs, one to extract features from an input image and the other to transpose convolution to create a prediction of how the object would look like in 3D, from the features extracted by the first neural network. RayNet[9] is a CNN that learns multi-view image similarity with an Markov Random Fields with ray potentials that explicitely models perspective projection and enforces occlusion constraints across viewpoints. The physics of multi-view geometry is embeded into RayNet. RayNet is not required to learn the complex relationships from data, instead the network focuses on learning view-invariant feature representations that are difficult to model.

3D Reconstruction of porous objects The typical approach to determine the geometry and topology of a pore space of a porous material is through the construction of a 3D model of the true material. Knowing the porosity factor of an object is beneficial for many areas such as the medical area, to improve the friction and wear behaviour of artificial hip joints or knee prostheses[8] by measuring a stepwise milled surface by fly cutting with a confocal laser scanning microscope; to 3D print patient-specific implants for people undergoing tumor removal and bone cancer treatment[13], the implants are either standardized in terms of shape and size or can take weeks to deliver if they are customized. Knowing the optimal porosity factor of a bone could be an advantage to create a 3D printed implant.
Other methods have been proposed to 3D reconstruct porous objects such as making use of Gaussian Random Fields(GRF) that accelerates the reconstruction process and creates rough large 3D models[3]; procedurally generating microstructures based on their elastic behaviours and Voronoi open-cell foams, the advantage of this method is that you can create objects with spatially varying elasticity and you can produce very detailed structures without having to explicitely produce a full representation of the complete object[6].

3 TECHNICAL BACKGROUND

A **convolutional neural network (CNN)** is a class of deep neural networks, most commonly applied to analyzing visual imagery. [reference Wikipedia] All the layers we will mention are the basic building blocks of any CNN. We will discuss some machine learning terms to refresh the mind.

Epoch: An epoch is one complete presentation of the data set to be learned to a learning machine.

ReLU: Stands for Rectified Linear Unit and is a non-linear operation. ReLU is an element wise operation and replaces all negative pixel values in feature map with zero Fig2.

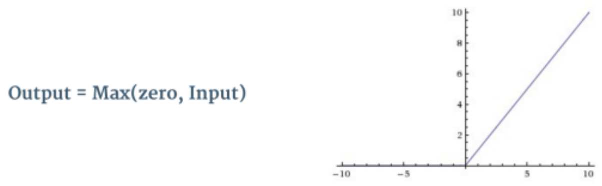


Figure 1: The ReLU operation. Source [14]

The purpose is to introduce non-linearity in our Convolutional network since most of the real-world data we would want out convolutional neural network to learn would be non-linear.

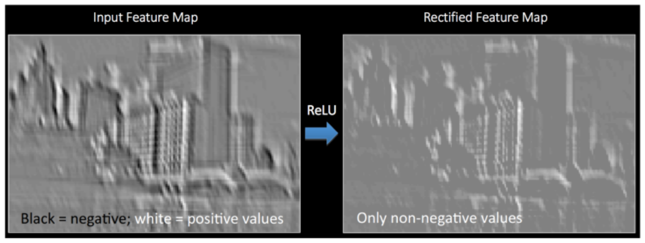


Figure 2: ReLU operation. Source [1]

Convolutional Layer: The convolutional layer extracts features from the input image. It perserves the spatial relationship between pixels by learning image features using small squares of input data. We know that an image is considered as a matrix of pixel values. Lets take an image I that is size of 7x7 whose pixel values are 0 and 1 (gray scale image) 3. Consider also a 3x3 matrix that is called a filter, K. If you slide that filter K over the image I and multiply the values you will get a feature map.

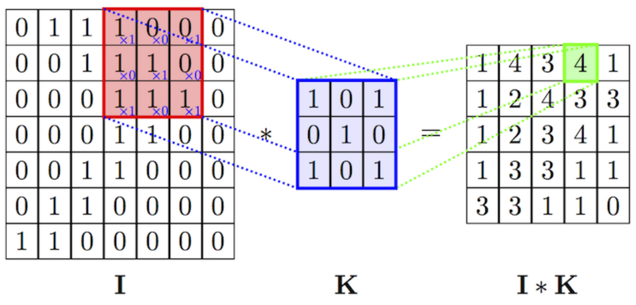


Figure 3: explained convolution. Source [15]

This is used for multiple applications such as edge detection, to sharpen or blur the image. Changing the numeric values of the filter matrix before the convolution operation will create different effects. Different filters can detect different features from an image such as edges, curves, etc. Basically CNN learns the values of these filters on its own during training process. The more number of filters we have, the more image features get extracted, the better our network becomes at recognizing patterns in unseen images.

Pooling Layer: Pooling, also called subsampling or downsamling reduces the dimensionality of each feature map but keeps the most important information. There are different types of pooling such as max pooling, average pooling and sum pooling. Lets define a 2x2 window and slide the window over the image. Every time you slide one step you take can take either the largest element Fig.4, the sum or the average within that window. Max pooling has shown to work better than the other two.

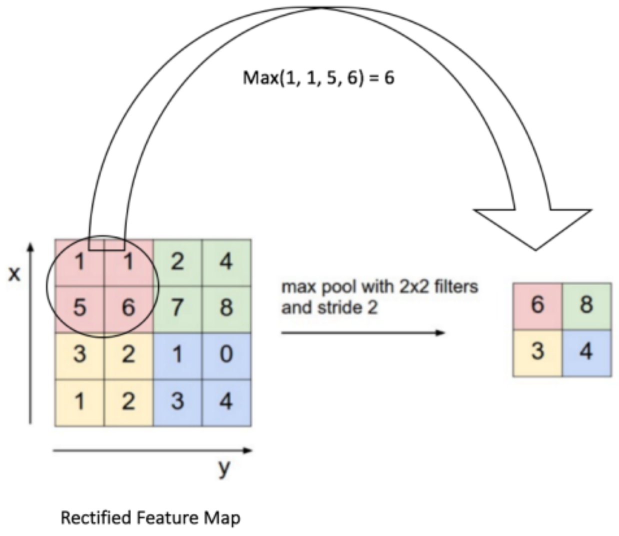


Figure 4: Max Pooling. Source [4]

The purpose of this is to progresively reduce the spatial size of the input representation. The benefits of pooling would be that the:

- input representation are smaller and manageable
- it reduces the number of parameters and computation in the network, therefore, controls the overfitting
- it makes the network invariant to small transformations, distortions and translations in the input image
- representation of our image would be almost be scale invariant, this is very powerful since we can detect objects in an image no matter where they are located.

Fully connected or Dense layer A fully connected or dense layer is a traditional multi-layer perceptron that uses a softmax activation function in the output layer. Every neuron in the previous layer is connected to every neuron on the next layer. The purpose of a fully connected layer is to use the the outputs from the convolutional and pooling layers for classigying the input image into various classes based on the training dataset. The Softmax function takes a vector of arbitrary real-valued scores and squashes it to a vector of values between zero and one that sum to one.

Drop out layer Dropout refers to ignoring neurons during the training phase of certain set of neurons which is chosen at random. Ignoring, means that the neurons are not considered during a particular forward or backward pass. This is to prevent over-fitting. A fully connected layer occupies most of the parameters, and hence, neurons develop co-dependency amongst each other during training which curbs the individual power of each neuron leading to over-fitting of training data.

4 METHOD OVERVIEW

To achieve a 3D reconstruction of the internal structure of a bone we had to follow a certain pipeline Fig.5. First we had to find a dataset that fits the criteria of our neural network. Based on how big the dataset is we would augment the images to create a larger dataset. Depending on what the neural network needs to learn we had to label each image in the dataset, labelling the images in a correct way is important, if it is not labelled in the correct way the network could predict something unexpectedly. When the dataset was ready to be fed into a neural network we had to create our neural networks. We then train the networks for a certain amount till we are satisfied with the result. Depending on their accuracy we would integrate our neural network models into our application that would 3D reconstruct the internal structure of our bone based on the porosity factor range prediction and its health factor Fig.5.

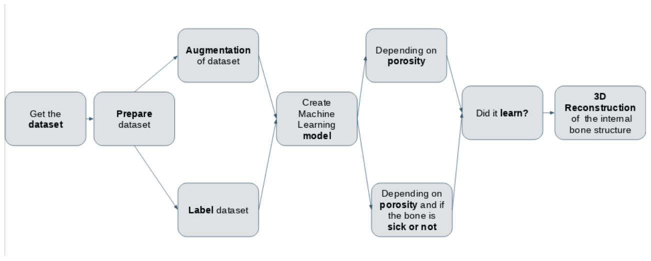

Get the dataset
Prepare dataset
Augmentation of dataset
Label dataset
Create Machine Learning model
Depending on porosity
Depending on porosity and if the bone is sick or not
Did it learn?
3D Reconstruction of the internal bone structure

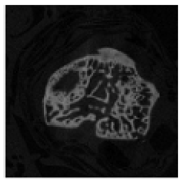
Figure 5: Pipeline for reconstructing internal bone structure

4.1 Augmentation of dataset

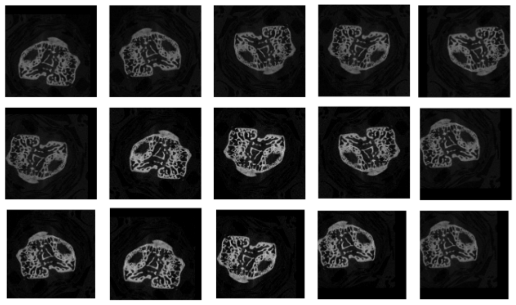
We used a dataset[11] that is composed of reconstructed 18.900 ct-scans of the proximal part of 21 tibia from wild-type mice, osteogenesis imperfecta mice and mice transplanted with human amniotic fluid stem cells.
For the **neural network depending on the health factor** we took all the images from the dataset excluding the mice transplanted with human amniotic fluid stem cells, this was an average of 10,800 images. A neural networks accuracy can depend on how big the dataset is, therfore we took the decision to augment the dataset. Consider the original image in Fig.6a, we would get the resulting images illustrated in Fig.6b, when the augmentation methods were applied to it. The way we augmented the images were by **flipping** them horizontally, vertically and a combination of both; Then we applied **translations** on them and varied their **brightness**. By doing the augmentation we ended with 190,291 images which is 17 times bigger than the dataset we originally had. To do the augmentation of the images we used three libraries, OpenCV, Python Imaging Library(PIL) and numpy. The below code illustrates what function we used from the libraries. As said before we flipped our images horizontally, vertically and a combination of both.

```
# flip original image
flipped_imgh = cv2.flip(img,0)
flipped_imgv = cv2.flip(img,1)
flipped_imgb = cv2.flip(img,-1)
```

When we applied translation to the image we had to define a range first on how far a image could translate. We added randomness to the translation of images. The problem of translating the image was that it would loose its original size, therefore we had to pad the image with black pixels till the image had its original size. Fortunately, OpenCV has a function for it, see code below.



(a) Original image



(b) Variations

Figure 6: Augmentation of dataset images

```
t_range = 300
# translate the image randomly on x- and y-axis
tr_x = trans_range*np.random.uniform()-t_range/2
tr_y = trans_range*np.random.uniform()-t_range/2
# create translation matrix
trans_M = np.float32([[1,0,tr_x],[0,1,tr_y]])
# pad matrix with black pixels
img = cv2.warpAffine(img, trans_M, (960,960))
```

We also varied the brightness on the images by randomly change the contrast. The first step would be to generate a random number every time we read an image and then enhance the contrast depending on that number.

```
randomNumb = round(random.uniform(0.3,2),1)
contrast = ImageEnhance.Contrast(img)
contrast = contrast.enhance(randomNumb)
```

For the **porosity factor dependant neural network** we trained it using the images from a mouse transplanted with human amniotic fluid stem cells, so we had a total of 21,000 images, after labelling them we realised the dataset was not uniform enough so we augmented some parts of the set and reduced other parts depending on their relative size. We applied all the augmentation methods mentioned above in addition with a new method, we created a rotation method that rotates the image based on a random angle. Using this method allowed for a more diverse range of images for the data set.

```
#calculate random angle of rotation
rotNum=random.randrange(1, 23)*15
# set rotation center to center of image
image_center = tuple(np.array(
                        img.shape[1::-1]) / 2)
#calculate rotation matrix
#based on "rotnum" angle
rot_mat= cv2.getRotationMatrix2D(image_center,
angle, 1.0)
#translate based on rotation matrix
#and store in result
result = cv2.warpAffine(img, rot_mat,
                        img.shape[1::-1],
                     flags=cv2.INTER_LINEAR)
```

Applying these augmentation methods resulted in a data set containing 1000 images for each porosity factor range, resulting in a total of 20000 images for the neural network to train on. Bellow is an example of image augmentation Fig. 7b

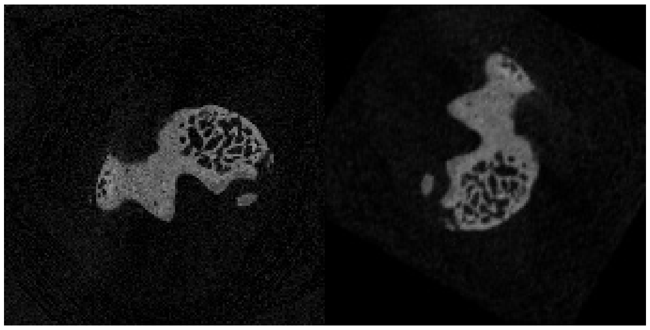

(a) Original image (b) Augmented Image

Figure 7: Example of image augmentation

4.2 Labelling the dataset

The next step was to label the dataset depending on what we wanted the neural network to learn. We decided to create two neural networks to get more information about the bone. The first CNN would depend on the health factor of the bone and the second CNN would depend on the porosity factor of the bone. The original dataset[11] was divided into three categories, and in those categories it was split into 6 different mice. For labelling the dataset based on the health factor we had to take all the images from the wild-type mice and the ostegonesis mice folders and put them respectively into a health and sick folder. Doing this did not take much time as it was basically already done for us. The core of our application was to reconstruct a porous object based on its porosity factor. Labelling the images depending on the porosity was challenging, as the area of interest was what was inside the outline of the bone. Our first approach was to get rid of the noise in the image by blurring it and then taking the binary threshold of it. We then create an alpha mask Fig.8b to represent the contour of the bone by using the contour function and if in the image there would remain holes in the mask we would use the morphing function (close) to fill up the holes. The final step was to add the first thresholded image with the alpha mask to create an image with a gray background Fig.8c that would make us able to calculate the porosity factor by getting the amount of black pixels and the total amount of pixels (white and black) Eq.1.

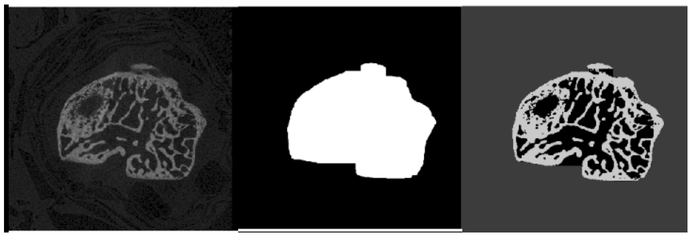

(a) Original image (b) Alpha mask (c) Final image

Figure 8: Process to get the porosity factor of the bone

$$porosity factor = \frac{black}{black + white} \tag{1}$$

We tried to find a 'correct' way to label images based on the porosity factor, but because our dataset was quite small we concluded to label the porosity factor in ranges that gave us 20 different classes instead of 100. The first folder(range), would go from 0 to 5 percent and the last folder(range), would go from 95 to 100 percent. We know that the way we labelled our dataset would not give us a precise porosity factor of the bone but we concluded that this would give us an good enough result.

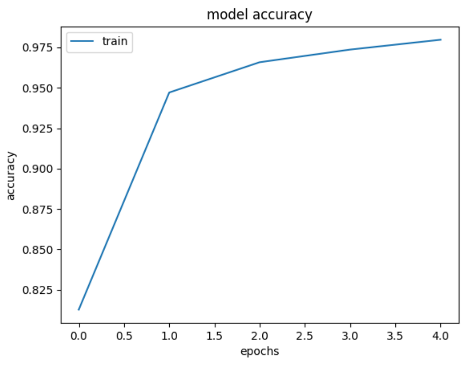
5 RESULTS

To train the neural network we split the dataset into a training and test set, the training set was 90 percent of the original dataset and the test set was 10 percent of it. We first stored all the images in a HDF5 file(.hdf5) with their labels, so we could load the data set whenever we wanted in different scripts, this was useful as there was no need to create the dataset every time we wanted to train or test our neural networks; creating the dataset every time took a lot of time compared to just loading up the dataset with a .hdf5 file. To split the dataset we used the function from the library Keras, train_test_split(images, labels, test_size).

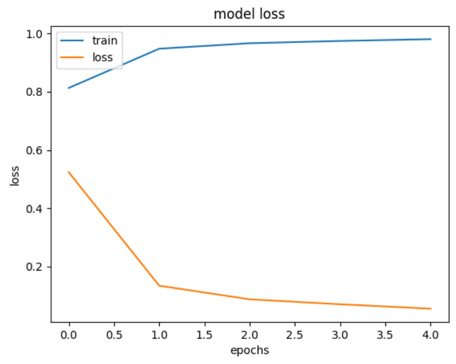
Neural network depending on health factor: The architecture of the neural network contained 4 convolutional layers, 3 pooling layers, 5 drop out layers, 3 dense layers and one layer that flattens all the data. Firstly we trained the neural network depending on the health factor and after 28.08 minutes and 5 epochs it had an accuracy of 97.98% and a loss of 0.0558. We thought the accuracy was high enough so we did not continue to train the neural network as it could overfit. Fig.12a illustrates that the accuracy gets better with the the amount of epochs. Idem for Fig.12b, the loss reduces with the amount of epochs.
We then tested our neural network by feeding it images that it had never seen before, and it gave an accuracy of **99.13%** and a loss of **0.029**, which is a very good evaluation. We ran the prediction method on a test data set and it predicted every image correctly except one. Fig10 illustrates 36 images, the caption under the image shows three variables, the first variable shows the prediction, the second variable shows the percentage of the prediction and the

last variable shows the real label of the image; the color of the caption displays if the prediction is correct(blue) or not(red). We were pleased with the neural network as it was successful of predicting the health factor of each slice.

Neural network depending on porosity factor: The first model we tested for the porosity factor prediction had an accuracy of 50% which is not good enough for our purposes. The model used was not appropriate so we modified it until we got a high enough consistent accuracy. The final model contained four convolutional layers, four max pooling layers, two dense layers and a dropout layer. Training it took 26 minutes for 50 epochs This combination resulted in a near perfect network with a 95% accuracy and a loss of 0.1 Fig.12. We then evaluated the network on a completely different data set and got a 95% accuracy rate. Finally we fed in a entire ordered set of bone cross sections to analyze the porosity variance across the bone and it resulted in the graph at Fig.11



(a) Accuracy with the amount of epochs



(b) Loss with the amount of epochs

Figure 9: Accuracy of model depending on health factor

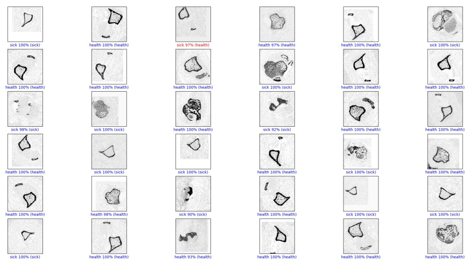


Figure 10: Prediction of neural network depending on health factor

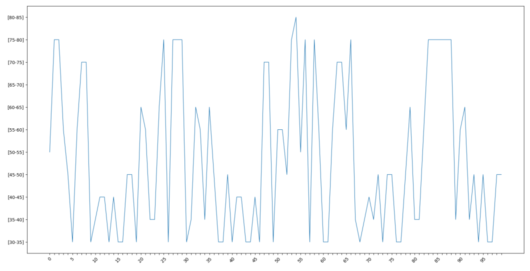


Figure 11: Graph showing the porosity factor variance throughout 100 cross sections

Reconstruction: Unfortunately, we did not reconstruct the bone based on its porosity factor because of time constraints, but we looked into the library openVDB to have a basic knowledge how to convert 2D images into 3D models. We created a script that would create a 3D grid, would iterate over each image slice of a bone and if the pixel value of the image was 255(white) it would fill a voxel based on its position of the 3D grid. This gave us a nice 3D display of the bone Fig13. For future work we suggest creating a neural network which analyses a sequence of images and determines if they are in order. This determination could pave the way for automatic reconstruction based on a couple of inputs such as the outline of a bone, the porosity factor, the healthiness of a bone and only a few cross sections.

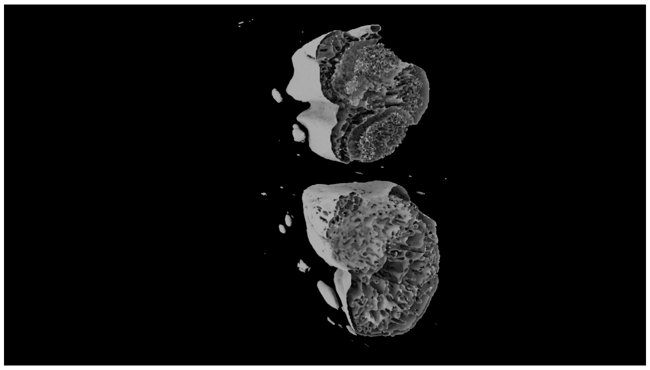
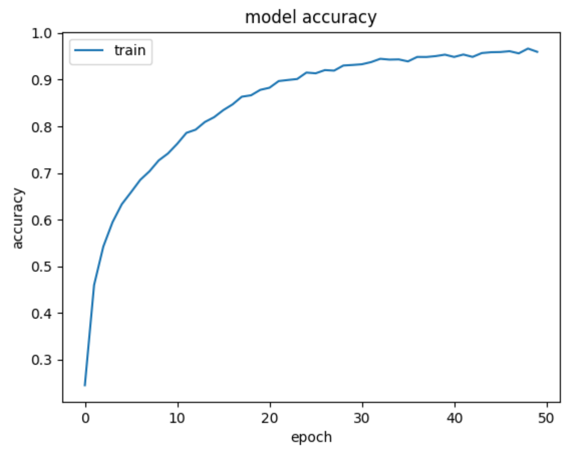
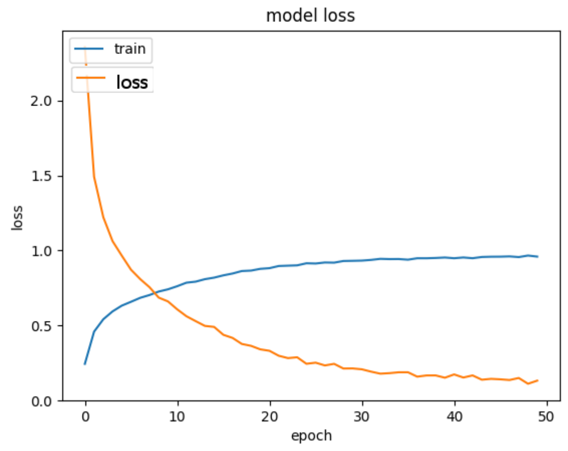


Figure 13: 3D models of sick bone(bottom of image) and healthy bone(top of the image), using our script but not based on porosity factor



(a) Accuracy by the amount of epochs



(b) Loss by the amount of epochs

Figure 12: Accuracy of model depending on porosity factor

6 CONCLUSION

We have proposed an approach to 3D reconstruct a porous object such as bone by using machine learning. The use of machine learning was to predict the porosity and health factor of the bone. We fed the neural networks images from a dataset[11] that we augmented in different ways to enlarge the dataset and labelled them according to what the neural network needed to learn. We successfully taught the neural networks the porosity factor ranges and the health factor of a bone slices respectively. We were pleased with the final results that came from our neural networks, the neural network depending on the health factor had an accuracy of 99.13% and the neural networks depending on the porosity factor had an accuracy of 95%.

Our future work would involve the reconstruction of the bone by creating a script that includes the two neural network models. First it predicts if the bone is sick or not by averaging the prediction of the health factor from all the slices. This is just to inform the user if the bone its health factor. Then the slices go through the model that predicts the porosity factor, and based on the porosity factor it reconstructs the bone procedurally with the help of an algorithm like Jonas Martinez et al. method[6]. The algorithm depends on two parameters and one of the parameters is the density, which we would replace with the porosity factor. Our application then creates a openVDB file (.vdb) that you then can upload in Houdini by creating a file node, load in the file and then convert the file into a polygon. The porosity factor network predictions were precise but since the porosity factor was based on discrete ranges as opposed to a continuous range it is not ultra precise. A possible solution to this would to find better image processing methods to calculate the porosity and find a bigger dataset to classify the images in 100 folders instead of ranges of 20 classes.

REFERENCES

[1] Rog Fergus. 2015. Neural Networks. *http://mlss.tuebingen.mpg.de/2015/slides/fergus/Fergus_1.pdf* (2015).

[2] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. 2015. Spatial Transformer Networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 2017–2025. http://dl.acm.org/citation.cfm?id=2969442.2969465

[3] Zhen Jiang, Wei Chen, and Craig Burkhart. 2012. A Hybrid Approach to 3D Porous Microstructure Reconstruction via Gaussian Random Field. *Proceedings of the ASME Design Engineering Technical Conference* 2. https://doi.org/10.1115/DETC2012-71173

[4] karpathy@cs.stanford.edu. [n. d.]. CS231n Convolutional Neural Networks for visual Recognition. *http://cs231n.github.io/convolutional-networks/* ([n. d.]).

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12)*. Curran Associates Inc., USA, 1097–1105. http://dl.acm.org/citation.cfm?id=2999134.2999257

[6] Jonàs Martínez, Jérémie Dumas, and Sylvain Lefebvre. 2016. Procedural Voronoi Foams for Additive Manufacturing. *ACM Trans. Graph.* 35, 4, Article 44 (July 2016), 12 pages. https://doi.org/10.1145/2897824.2925922

[7] AF Mavrogenis, R Dimitriou, J Parvizi, and GC Babis. 2009. Biology of implant osseointegration. *J Musculoskelet Neuronal Interact* 9, 2 (2009), 61–71.

[8] Eduard Reithmeier Nina Loftfield, Markus Kastner. 2017. 3D Reconstruction And Characterization Of The Porous Microstructure Of $AL_2O_3 - CoatingsBasedOnSurfaceData.$(2017).

[9] Despoina Paschalidou, Ali Osman Ulusoy, Carolin Schmitt, Luc Gool, and Andreas Geiger. 2018. RayNet: Learning Volumetric 3D Reconstruction with Ray Potentials. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society.

[10] KIMBERLY POWELL. 2017. Deep Learning Opens Door to Intelligent Medical Instruments. *https://blogs.nvidia.com* (2017).

[11] Anna Maria Ranzoni. 2018. Micro-computed tomography reconstructions of tibiae of stem cell transplanted osteogenesis imperfecta mice. *figshare* (2018).

[12] Patrik Stigeborn. 2018. Generating 3D-objects using neural networks. *http://www.diva-portal.org/smash/get/diva2:1218064/FULLTEXT01.pdf* (2018).

[13] Tess. 2017. $9.3M Just in time 3D printed bone implant project in Australia set to transform tumour surgery. *https://orthofeed.com/2017/10/30/9-3m-just-in-time-3d-printed-bone-implant-project-in-australia-set-to-transform-tumour-surgery/* (2017).

[14] ujjwalkarn. 2016. An Intuitive Explanation of Convolutional Neural Networks. *https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/* (2016).

[15] Anh Vo. 2018. Deep Learning - Computer Vision and Convolutional Neural Networks. *https://anhvnn.wordpress.com/2018/02/01/deep-learning-computer-vision-and-convolutional-neural-networks/* (2018).

[16] Kai Wang, Manolis Savva, Angel X. Chang, and Daniel Ritchie. 2018. Deep Convolutional Priors for Indoor Scene Synthesis. *ACM Trans. Graph.* 37, 4, Article 70 (July 2018), 14 pages. https://doi.org/10.1145/3197517.3201362

[17] Meng Wang, Lingjing Wang, and Yi Fang. 2017. 3DensiNet: A Robust Neural Network Architecture Towards 3D Volumetric Object Prediction from 2D Image. In *Proceedings of the 25th ACM International Conference on Multimedia (MM '17)*. ACM, New York, NY, USA, 961–969. https://doi.org/10.1145/3123266.3123340