Spine Image Segmentation & Classification
ELL888 Assignment 1

Saksham Jain
2017MT10747
mt1170747@iitd.ac.in

Sumanth Varambally
2017MT60855
mt6170855@iitd.ac.in

Ritvik Sharma
2017EE10485
ee1170485@iitd.ac.in

Abstract

Spinal cord injuries suffered due to trauma need immediate attention from a medical expert. Early detection is imperative to improve the outcome of treatment and rehabilitation. This project aims to utilise the recent advances in Machine Learning, specifically Convolutional Neural Networks, to perform the task of segmentation and classification of spinal X-ray images as normal or damaged. We use a modified version of the U-Net architecture for the segmentation task, and use transfer learning with a Resnet50 mode and combining the resultant outputs to train an FCNN on top for classification to achieve satisfactory results. Additionally, we examine our model using SHAP (Shapley Additive Explanations) to verify that our model learns useful features from the input image.

1 Introduction

A spinal cord injury (SCI) is damage to the spinal cord that causes temporary or permanent changes in its function. Symptoms may include loss of muscle function, sensation, or autonomic function in the parts of the body served by the spinal cord below the level of the injury. In the majority of cases the damage results from physical trauma such as car accidents, gunshot wounds, falls, or sports injuries, but it can also result from nontraumatic causes such as infection, insufficient blood flow, and tumors. Spinal column injuries can be fatal, and need immediate attention. The process of isolating different parts of the spine and detecting the damage can be cumbersome and prone to error, even by trained professionals. [1]

We are given Anteroposterior(AP) and Lateral(LAT) views of the spinal cord (X-ray images), and we have to label the given spine as being 'Damaged' or 'Normal'. The training data consists of Anteroposterior (AP) and Lateral (LAT) views along with the corresponding Segmentation masks (AP Pedicle, AP Spinous Process, AP Vertebra for AP view, and LAT Anterior Vertebral Line, LAT Disk Height, LAT Posterior Vertebral Line, LAT Spinous Process, LAT Vertebra for the LAT view). The AP and LAT view images are 3-channel RGB images (.jpg) and the corresponding masks are one-channnel binary images (.png). The dataset has 328 training examples for the class 'Damaged', and 350 training examples for 'Normal'.

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

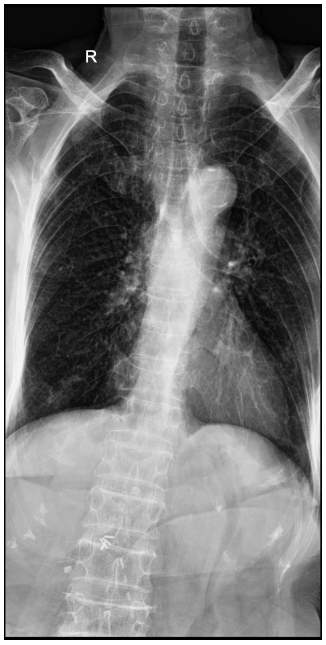

Figure 1: Sample AP Image

Figure 2: Sample LAT Image

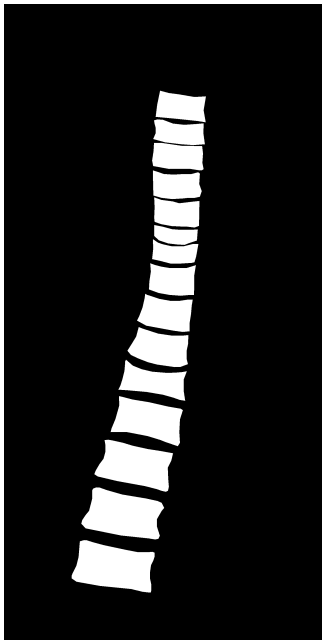

Figure 3: Sample Segmentation Mask 1

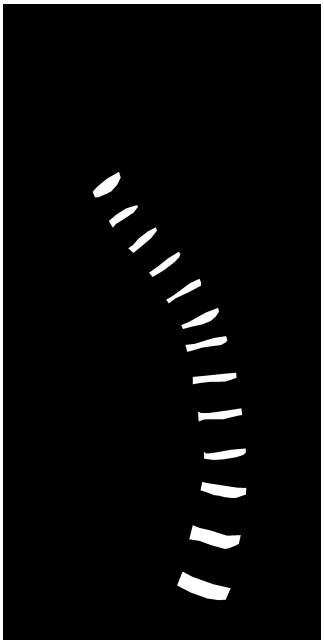

Figure 4: Sample Segmentation Mask 2

1.1 Exploratory Data Analysis (EDA)

PCA (Principal Component Analysis) was applied to the training data, to see the dependence of cumulative variance over the number of features. The same has been plotted in the given figure. Upon taking the top two components (corresponding to the largest eigenvalues), the datapoints (now in two dimensions) were plotted for each class separately (blue dots correspond to 'Normal', while red dots are for 'Damaged').

There did not seem any noticeable visual separation between the two components.
t-SNE (t-Distributed Stochastic Neighbor Embedding) was also employed for data exploration, whose results are plotted in the given figures.
It is evident that t-SNE is somewhat successful in separating out the two classes (compared to PCA). This difference in performance can be explained by the fact that t-SNE preserves only small pairwise distances or local similarities, whereas PCA is concerned with preserving large pairwise distances to maximize variance.[2]
For the task in hand, maybe the relative spatial arrangement of different bones in vicinity is more important in determining the label (Normal/Damaged), and far-away dependencies are not of much statistical importance.

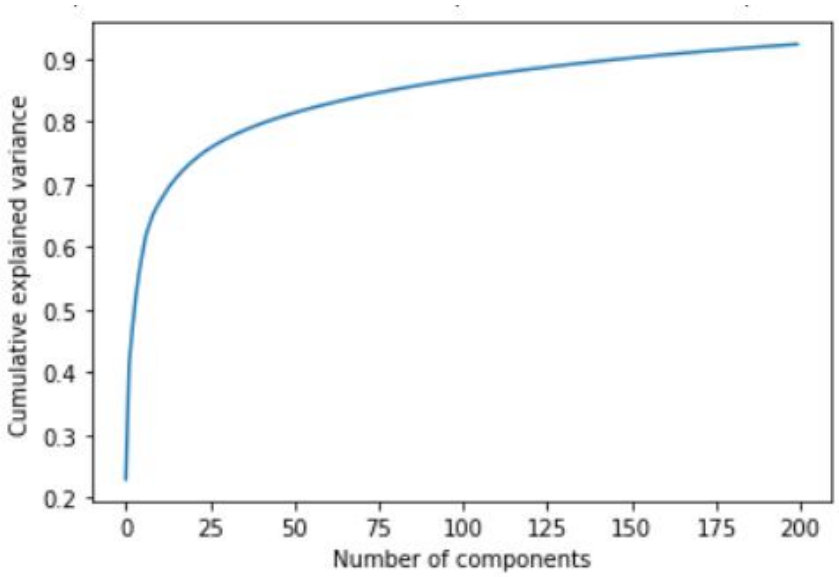

Figure 5: PCA Variance Capture

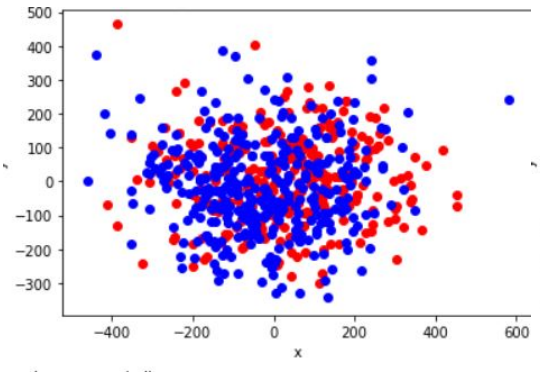

Figure 6: Top two components of PCA (AP)

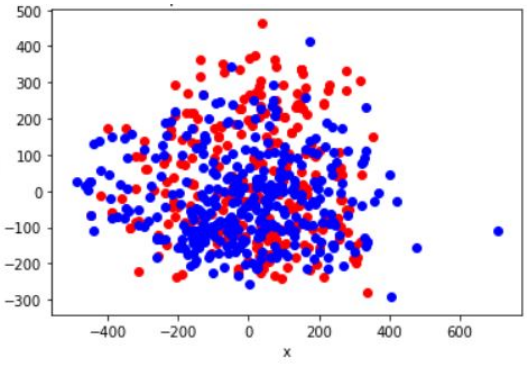

Figure 7: Top two components of PCA (LAT)

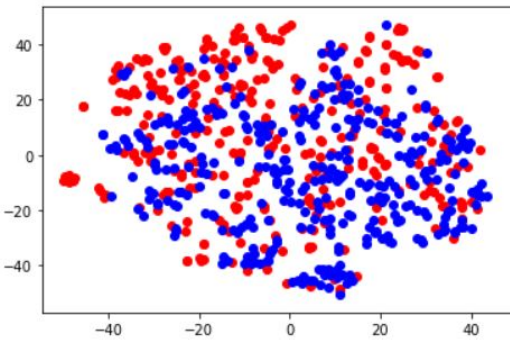

Figure 8: t-SNE (2 Dimensions) (AP)

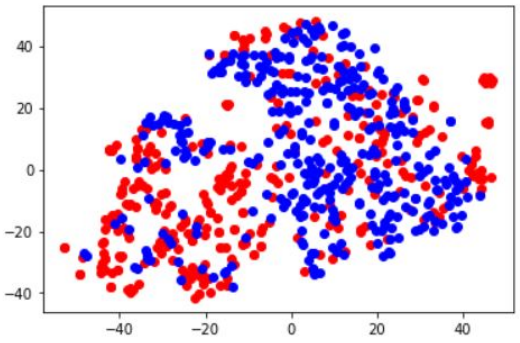

Figure 9: t-SNE (2 Dimensions) (LAT)

2 Segmentation Architecture

Image segmentation is a pixel-level classification task. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. It makes analysis and further processing of the image easy[7]. There are two types of segmentation tasks: Semantic Segmentation and Instance Segmentation. In semantic segmentation, the different objects belonging to the same class are considered the same entity, and hence, assigned the same label. Instance segmentation is more robust, and is able to identify different objects belonging to the same class as differently.[3]
The given problem is an instance segmentation problem, where corresponding to each Anteroposterior(AP) and Lateral(LAT) view, we have to segment out the relevant masks. We have used U-Net for the purpose. The U-Net was developed by Olaf Ronneberger et al. for Bio Medical Image Segmentation. The architecture contains two paths. First path is the contraction path (also called as the encoder) which is used to capture the context in the image. The encoder is just a traditional stack of convolutional and max pooling layers. The second path is the symmetric expanding path (also called as the decoder) which is used to enable precise localization using transposed convolutions. Thus it is an end-to-end fully convolutional network (FCN), i.e. it only contains Convolutional layers and does not contain any Dense layer because of which it can accept image of any size. [4]

In this assignment, a slightly modified version of U-Net has been employed. Firstly, the upsampling layers have been replaced with transposed convolutional layers (Conv2DTranspose). When the traditional U-Net architecture from [9] was tested, the model quickly stopped learning. We suspect that replacing the upsample layers with the transposed convolutional layers lowered the chances of the vanishing gradient issue. The inspiration for this idea came from the implementation from [8].

Another modification we employed was to enable the U-Net architecture to output multiple masks at once. From the last convolutional layer, we branched out into three masks (in the case of AP view) and five masks (in the case of LAT view). Thus, we trained two seperate networks for the segmentation task - one trained on the AP images and another on the LAT images. The motivation behind this was that the information captured in the images of one view was shared among the predicted masks, but the two views were pretty different from one another. Thus, they were treated as seperate tasks and dealt with accordingly.

	Dice	Jaccard	False Positive	False Negative
AP Pedicle	0.747	0.611	0.003	0.1775
AP Spinous Process	0.419	0.2788	0.0007	0.5795
AP Vertebra	0.867	0.7744	0.0128	0.1205
LAT Anterior Vertebra Line	0.079	0.043	0.0025	0.8796
Lat_Disk_Height	0.7739	0.644	0.0047	0.203
Lat_Posterior_Vertebral_Line	0.0875	0.047	0.0026	0.8512
Lat_Spinous_Process	0.7765	0.65	0.0089	0.1838
Lat_Vertebra	0.8715	0.782	0.0101	0.1189

Figure 10: Results

2.1 Data Augmentation

For the data segmentation task, we employed data augmentation to enable better performance, since only roughly 650 samples were available for training. This was performed using the Augmentor library [10]. The operations employed were random distortions, skewing, rotations and random zooming. One thing to note was that even though these transformations changed the nature of the X-ray images, i.e. a normal X-ray could be considered a damaged one after these transformations, the goal was to enable the network to learn how to annotate different parts of the spine even under unrealistic scenarios (since the end-goal was to perform segmentation). To that end, normal and damaged spinal images were not segregated during the entire training process for the segmentation task.
As we can observe from the above results, the biggest problem with our models is the high False Negative Rate. This could be combated using asymmetric loss functions that specifically penalise high false negative rates.

3 Classification Architecture

First we tried CNN trained on one of the views. This approach trained a CNN on AP view to perform binary classification. This method is unable to take into account two images. We did not attempt to concatenate the images since they are two views of the same object and concatenating them in any manner will distort the CNN as they are not a single 2d image of an object that CNN assumes it to be.

To overcome this issue we can use CNNs on both images separately and then use a polling system which gives different priority to prediction by AP and LAT models. However we can extend this idea of giving weightage to these models to a general 2 input function and the nonlinear function can be approximated by an FCNN. So we decide to use CNN to encode the image to its features and then use an ANN as a universal function approximator to learn the weightage it should give to the various features in both the images and what weightage should be given for them. Now training a deep network with single input like above is difficult and hence we resort to transfer learning for feature extraction. This is followed by dense layers that are concatenated together to allow FCNN to learn a function that captures the weightage to be given to these features for training.
We tried multiple backbone feature extractors like Resnet50, VGG16 and inceptionV3. We plot the T-SNE of the output of these models for the train data. The T-SNE of the InceptionV3 in 2d looks indistinguishable. Whereas Resnet and VGG16 show better grouping. After experimenting with both we conclude that Resnet 50 trained on imagenet provided better results.

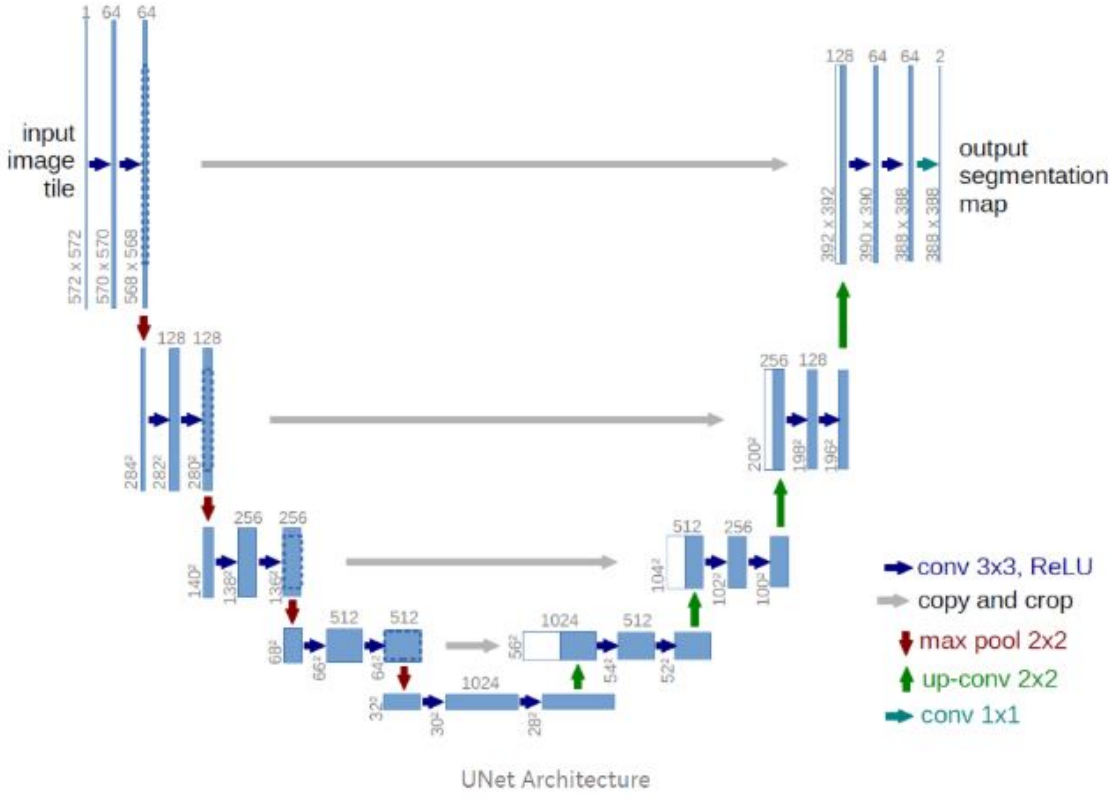

Figure 11: Standard U-Net Architecture [5]

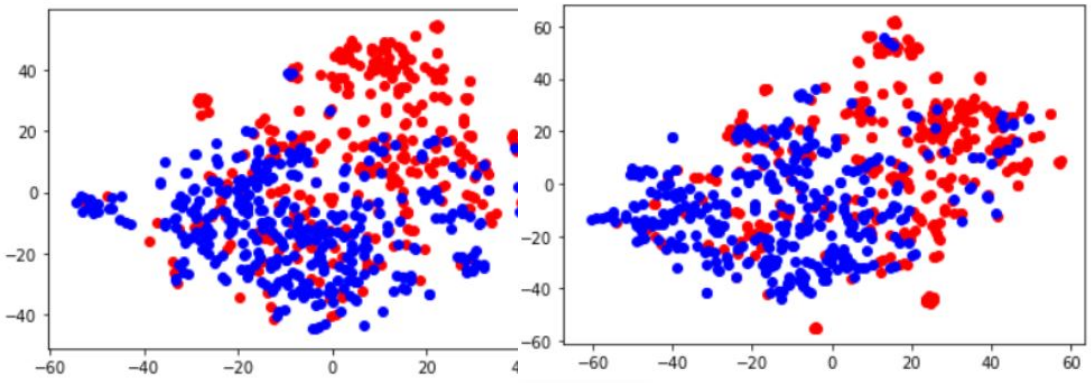

Figure 12: t-SNE for Resnet AP (2 Dimensions)

Figure 13: t-SNE for Resnet LAT (2 Dimensions)

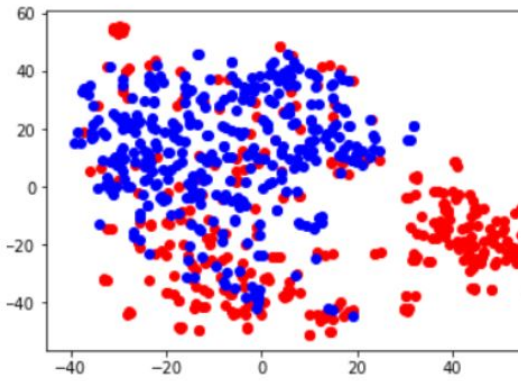

Figure 14: t-SNE for VGG AP (2 Dimensions)

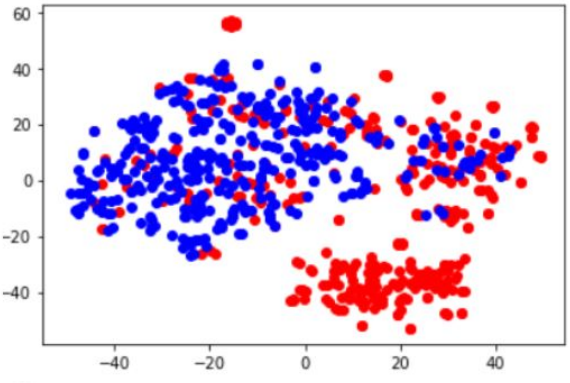

Figure 15: t-SNE for VGG LAT (2 Dimensions)

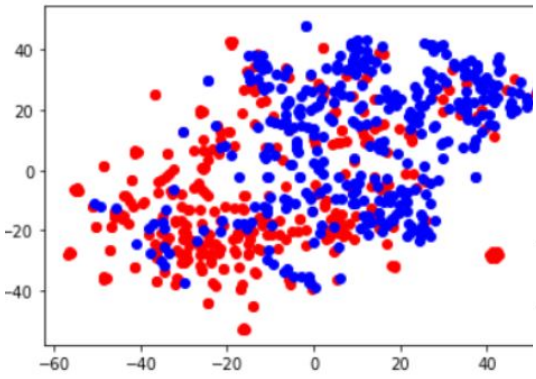

Figure 16: t-SNE for Inception Net AP (2 Dimensions)

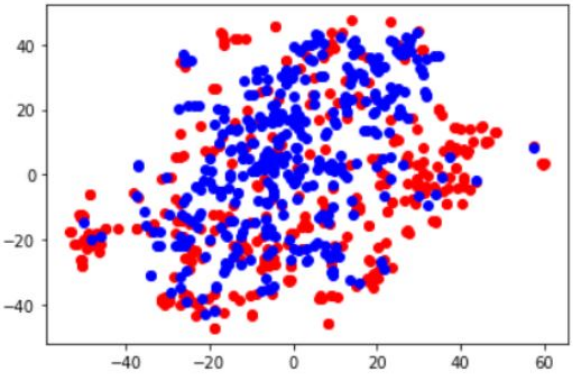

Figure 17: t-SNE for Inception Net LAT (2 Dimensions)

Resnet is a Convolutional Neural Network, with 'identity shortcut connections', which skip one or more layers. One motivation for skipping over layers is to avoid the problem of vanishing gradients, by reusing activations from a previous layer until the adjacent layer learns its weights. During training, the weights adapt to mute the upstream layer, and amplify the previously-skipped layer.[6]

The complete model architecture (for final classification task, using Resnet) is as shown in the image.

In the initial classification architecture, 256 neurons were put in the first dense layer, and error (training and validation) plotted against the number of epochs. The plot suggested an overfit, hence, number of neurons were reduced to 128 in the subsequent architecture. Further to reduce overfitting, the training was clipped at the 15th epoch, after observing training and validation losses vs number of epochs. Finally, an overall accuracy of 87% was obtained, with a batch size of 4 and Adam optimizer with learning rate of 1e-4 .

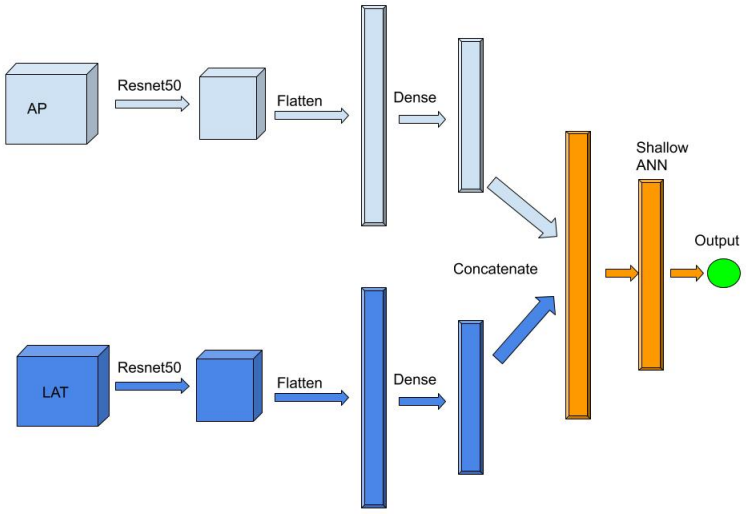

Figure 18: Complete Classificaion Architecture

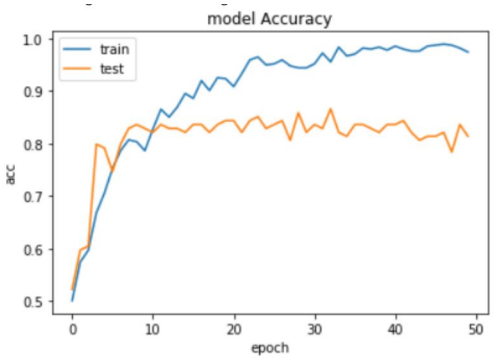

Figure 19: Plot of accuracy vs epochs for 256 Neurons

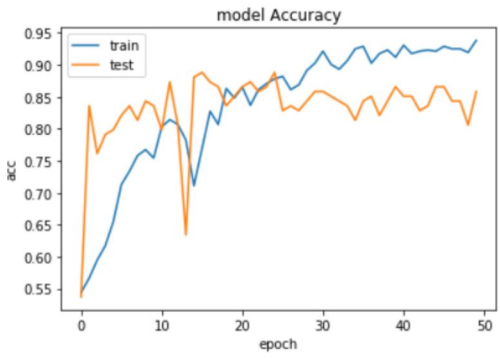

Figure 20: Plot of accuracy vs epochs for 128 Neurons

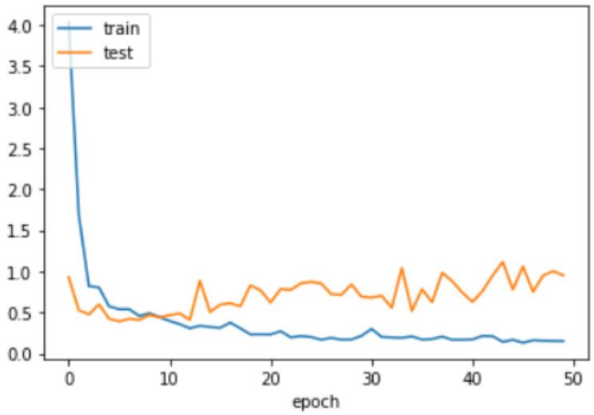

Figure 21: Plot of loss vs number of epochs

Classification Report

	precision	recall	f1-score	support
Normal	0.83	0.92	0.87	52
Damaged	0.90	0.79	0.84	48
accuracy			0.86	100
macro avg	0.87	0.86	0.86	100
weighted avg	0.86	0.86	0.86	100

Figure 22: Classification Results (100 samples)

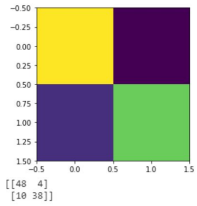

Figure 23: Confusion Matrix for Classification (100 samples)

4 References

[1] https://en.wikipedia.org/wiki/Spinal_cord_injury

[2] https://www.biostars.org/p/295174/

[3] https://datascience.stackexchange.com/questions/52015/what-is-the-difference-between-semantic-segmentation-object-detection-and-insta

[4] https://en.wikipedia.org/wiki/U-Net

[5] https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

[6] https://en.wikipedia.org/wiki/Residual_neural_network

[7] https://www.analyticsvidhya.com/blog/2019/04/introduction-image-segmentation-techniques-python

[8] https://github.com/jocicmarko/ultrasound-nerve-segmentation

[9] https://github.com/zhixuhao/unet

[10] Marcus D Bloice, Peter M Roth, Andreas Holzinger, Biomedical image augmentation using Augmentor, Bioinformatics https://doi.org/10.1093/bioinformatics/btz259