Bone Fracture Diagnosis on X-Ray Images Using CNNs

Walter Lizardo Priyanka Aiyer Abhav Vohra

Department of Electrical and Computer Engineering
NYU Tandon School of Engineering
wl30@nyu.edu pa2424@nyu.edu av3290@nyu.edu
Link to Github repository: https://github.com/wlizardo/xray-facture-diagnosis

Abstract

Radiology is a vital diagnostic tool, giving essential information for routine injury and disease prevention and evaluation. Radiographic imaging, such as X-ray, is one of the most popular image modalities to help provide physicians with prompt assessment of the patient since it can help visualize the interior of the patient's elbow in a timely manner. Traditionally, it takes a physician years of training in order to diagnose fractures from radiographic images. In recent years, thanks to the thrive of deep learning, a model that can classify and detect of different types of bone fractures can be trained within hours from annotated images to help diagnose elbow fractures. Also, Increased demand in radiology work causes workload errors in diagnosis and delay in results. We aim to make it cost effective and provide radiologists and technicians with a quick and reliable analysis on an X-ray. Leveraging a dataset of 14,863 Xray images, encompassing fractured and non-fractured bones from the publicly available "MURA" dataset, our research evaluates the effectiveness of Convolutional Neural Networks (CNN) architectures in detecting fractures.

Introduction

Artificial intelligence (AI) in radiologic diagnostics is well established,1–3 with predictive power in cases like fracture detection exceeding human performance.4 5 In the foreseeable future, broad utilisation of available and affordable models on simple, yet decisive problems will likely contribute to general diagnostic accuracy of imaging studies and thus improve the standard of care.The task of fracture detection in computer vision can be expressed as correctly classifying a radiographic image into fracture and nonfracture classes by detecting fracture features in the image.

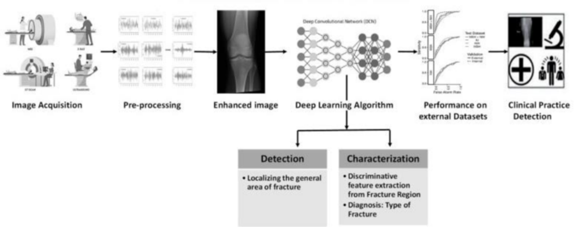
This project would focus on the set of X-ray images, in order to detect if there is any crack in the bones. Basically, the MURA dataset would be used for fracture detection, as it is one of the largest image datasets that contains seven classes of X-ray images classified in to 2 distinct classes; namely, "fractured" and "not fractured". MURA is a dataset of musculoskeletal radio-graphs consisting of 14,863 studies from 12,173 patients, with a total of 40,561 multi-view radiographic images. Each belongs to one of seven standard upper extremity radio-graphic study types: elbow, finger, forearm, hand, humerus, shoulder, and wrist. Each study was manually labeled as normal or abnormal by board-certified radiologists from the Stanford Hospital at the time of clinical radio-graphic interpretation in the diagnostic radiology environment between 2001 and 2012.[4]. In this project, we would train a number of CNN architectures, that is, basic CNN, ResNet, DenseNet, U-Net to figure out which model architecture is more adaptable to our problem.

Related Work

In a study by [3], a model was pre-trained on 100,855 bone images of several other body parts and subsequently fine-tuned using a Deep Convolutional Neural Network (DCNN) for detecting wrist fractures. In the study by [1], they evaluated VGG16, VGG19, DenseNet121, and DenseNet169 models in detecting and classifying humerus fractures. In 2018, Chung et al. developed a deep convolutional neural network to classify fracture types and achieved promising results with top-1 accuracy ranging from 65% to 86%. Negrillo et al., in 2020, proposed a geometrically-based algorithm for detecting landmarks in the humerus to reduce supracondylar fractures. They measured the distance between corresponding landmarks, finding a significant difference (1.45 mm, $p < 0.01$). Sezer et al. evaluated shoulder images using CNN for feature extraction and classified the head of the humerus into three categories: normal, edematous, and Hill-Sachs lesion, achieving an accuracy of 98.43%. Negrillo et al., in 2019, employed a geometrical and spatial approach to detect landmarks on the distal humerus, calculating six points for each bone. While some researchers have focused on humerus fractures, others have explored fractures in different bones such as the shoulder, femur, and calcaneus. De Vries et al., in 2021, worked on predicting the risk of osteoporotic fractures (MOF) and developed three machine learning models (Cox regression, RSF, and ANN). They found that Cox regression outperformed the other models with a concordance-index of 0.697.

Methodology

We plan to work with various different CNN architectures to find out the best suitable technique.


Deep Convolutional Network (DCN)
Image Acquisition
Pre-processing
Enhanced image
Deep Learning Algorithm
Performance on external Datasets
Clinical Practice Detection
Detection
• Localizing the general area of fracture
Characterization
• Discriminative feature extraction from Fracture Region
• Diagnosis: Type of Fracture

CNN Model Architecture

As the Convolutional Neural Network (CNN) is the widely used approach for image recognition and classification problems, we have initially designed a CNN model, in Python using Keras, to detect an abnormality in the input images. On this CNN model, the training and validation was performed. As the first layer, Convolution 2D was used to produce a tensor of outputs. The activation function 'ReLU' was then applied to remove linearity. An Adam optimizer was used as it involves a combination of two gradients' decent methodologies to give better results in terms of optimization. Moreover, the categorical cross-entropy was used to compute the training and validation loss. Subsequently, Max-pooling was used for down-sampling the spatial dimensions of input. Dropout was then applied to the network to prevent over-fitting. The output of the convolution layers was then flattened to generate a single feature vector, which are then linked to a fully connected layer as in a convolution neural network, all hidden layers are fully connected with each other.

ResNet Model Architecture

The ResNet neural network architecture allows for the creation of much deeper networks than traditional ones. This is achieved through the use of residual connections, which help to overcome the vanishing gradient problem that can occur when training networks with many layers. By allowing the network to be deeper, ResNet can learn more complex and abstract features from input data. These skip connections enable information to flow more effectively both forward and backward through the network. This helps the model to learn useful features at various levels of abstraction. By focusing on residual mappings, which are generally easier to optimize, while still benefiting from the identity mappings provided by skip connections, ResNets can achieve better performance than other architectures.

We decided to limit the number of layers in our implementation to decrease the number of trainable parameters. The initial convolutional layer has a kernel size of 7x7, a stride of 2 and padding of 3. This is followed by batch normalization and ReLU activation. The layer has 64 filters, resulting in 64 feature maps.

Our model consists of four stages, each containing a sequence of residual blocks. Each residual block contains two convolutional layers with 3x3 kernels, followed by batch normalization and ReLU activation. Shortcut connections are added to the residual blocks to handle different input and output dimensions. The number of filters in the residual blocks increases with each stage: 64, 128, 256, and 512.

After the final residual block stage, global average pooling is applied to reduce the spatial dimensions of the feature maps. This collapses each feature map into a single value by taking the average of all values in the map. The output of global average pooling is passed through a fully connected layer with a single output unit (for binary classification) with sigmoid activation. The model is trained using binary cross-entropy loss and optimized using the Adam optimizer.

DenseNet Model Architecture

The DenseNet (Densely Connected Convolutional Network) architecture is a deep learning model introduced by Huang et al. in 2017 that emphasizes dense connectivity between layers. A dense block consists of multiple densely connected layers, where each layer receives feature maps from all preceding layers within the same block. This dense connectivity means that the output of each layer serves as input to all subsequent layers in the block.Dense connectivity promotes feature reuse and enhances information flow throughout the network. By densely connecting layers, each layer has access to a rich set of features from all preceding layers, which can lead to more discriminative feature representations and efficient gradient propagation during training.To manage the computational complexity and enhance model efficiency, DenseNet incorporates bottleneck layers within each dense block. These bottleneck layers typically include 1x1 convolutional layers followed by 3x3 convolutional layers. The 1x1 convolution reduces the number of input feature maps before the 3x3 convolution is applied, controlling the growth of feature maps within the dense block.Between dense blocks, DenseNet employs transition layers to downsample feature maps and reduce the dimensionality. Transition layers typically include a combination of 1x1 convolutional layers for feature reduction followed by average pooling to downsample spatial dimensions. This helps in managing the growth of parameters and controlling overfitting.At the end of the network, global average pooling is often used to aggregate spatial information into a single vector, followed by a fully connected layer with softmax activation for classification tasks. This final layer produces the output probabilities for different classes based on the input image.

U-Net Model Architecture

U-Net is a convolutional neural network (CNN) architecture specifically designed for semantic image segmentation tasks, with notable applications in medical imaging but also relevant across diverse domains. Introduced in 2015 by Ronneberger et al., U-Net is characterized by its distinctive U-shaped structure, featuring an encoder (contracting path) followed by a decoder (expansive path). The encoder employs convolutional and pooling layers to progressively extract hierarchical features and reduce spatial dimensions, while the decoder uses upsampling and convolutional layers to restore the original image resolution. What sets U-Net apart is its incorporation of skip connections, which concatenate feature maps from the encoder to the corresponding decoder layers. These skip connections facilitate precise localization and detailed boundary delineation by preserving fine-grained information from earlier layers. The final layer of the U-Net architecture produces a segmentation map where each pixel's value represents the probability of belonging to a specific class, making it particularly effective in scenarios with limited training data. U-Net's flexibility allows for customization through adjustments in layer configurations, filter sizes, and skip connection strategies, making it adaptable to various segmentation challenges in both medical and non-medical contexts. In this task, we have modified the base U-Net Architecture to perform a binary classification task to identify fractures on x-rays.

Results

On implementing different CNN Architecture models like CNN, RestNet50, DenseNet and U-Net, we could achieve better performance and test accuracy on the MURA-V1.1 dataset.

We could achieve the desired goal, by using appropriate optimizers, augmentation methods, and fine-tuning the hyper parameters. Initially, we used the CNN model with which we could limit the total count of parameters to less than 1M parameters and for the ResNet50 model, we could limit the total parameters count to 7M parameters. Furthermore, on continuously experimenting with the improved versions of CNN architectures, along with different optimizers, augmentation methods, and fine-tuning the hyper parameters, we could train the CNN Model to attain better accuracy.

Training the ResNet model from scratch was a very time-consuming and resource-intensive process. After the first training epoch, it achieved an accuracy of over 60%. However, we were unable to train it for a longer duration. After three epochs, the validation accuracy did not improve significantly. We wished to investigate further, but the time-consuming process made it difficult to iterate over hyper-parameter settings. This experience emphasized the benefits of using a pre-train model.

A detailed summary of how the different models performed is mentioned in table 1.

Model	Accuracy
CNN	64.46
ResNet	63.62
U-Net	62.99
DenseNet Wrist	84.22
DenseNet Humerus	78.47
DenseNet Elbow	80.7
DenseNet Finger	64.8
DenseNet Forearm	67.1
DenseNet Hand	64.37
DenseNet Shoulder	74.77

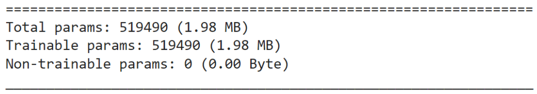
Table 1: Model architecture - Accuracy

Conclusion

In this project, we used different CNN architectures on the MURA-V1.1 dataset to detect if there is any abnormality in the bones. The designed models were used for fracture detection that contained seven classes of X-ray images classified in to 2 distinct classes; namely, "fractured (positive)" and "not fractured (negative)".

This project presented a detailed implementation of the CNN architecture Models using Keras and PyTorch for image classification tasks. In the initial CNN model, it was observed that it performed better as compared to other complex architectures in terms of training and validation accuracy. The initial CNN architecture model achieved a Training Accuracy of 85% and Validation Accuracy of 65% It was observed that when we train this CNN Model with more number of epochs, the Training as well as Validation Accuracy shows improvement in accuracy level. U-Net while supposedly effective in image segmentation, did not offer promising results for classification. Similarily, Resnet architecture also was not that effective for our problem statement. By far, DenseNet architecture while computationally expensive was considerable effective in accurately predicting fractures apart from other models tested.

- Number of parameters of the initial CNN Model

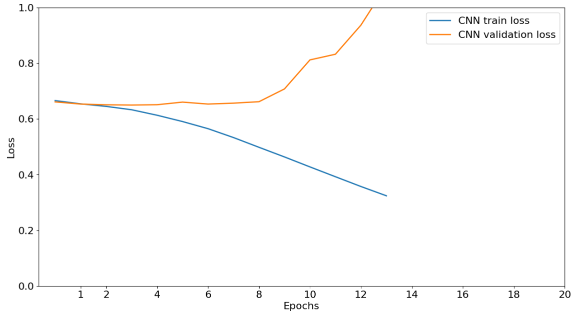


```
=================================================================
Total params: 519490 (1.98 MB)
Trainable params: 519490 (1.98 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

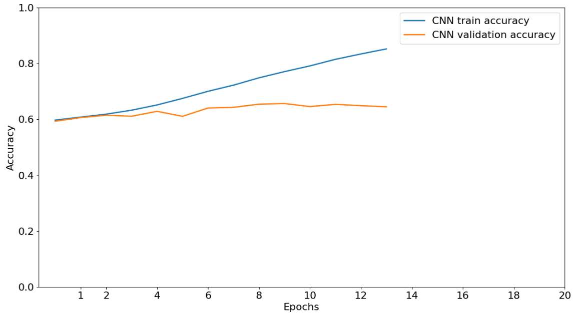
- Plotting Graphs of all Models

- Plotting Graphs: CNN model (MURA V1.1 dataset)

- Plotting train vs validation loss per epoch



- Plotting train vs validation accuracy per epoch



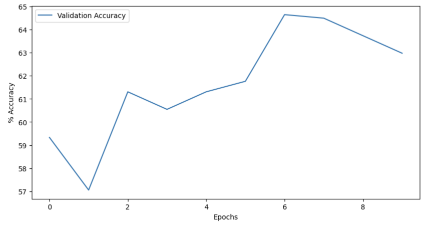


Figure 1: UNET

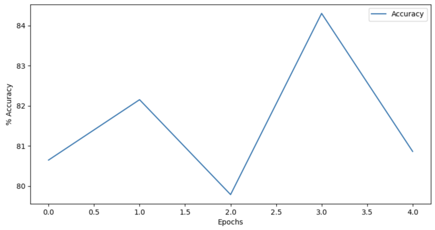


Figure 2: Elbow DenseNet

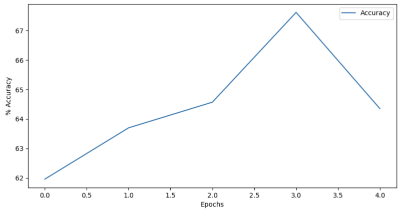


Figure 3: Hand DenseNet

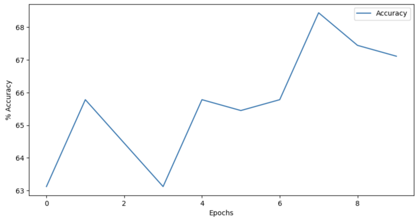


Figure 4: Finger DenseNet

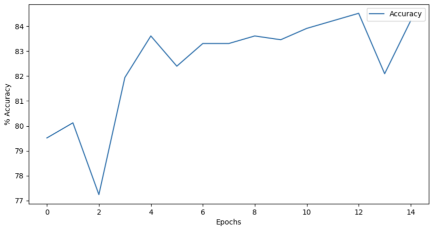


Figure 5: Wrist DenseNet

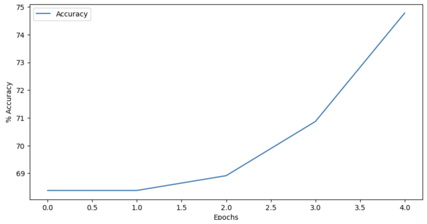


Figure 6: Shoulder DenseNet

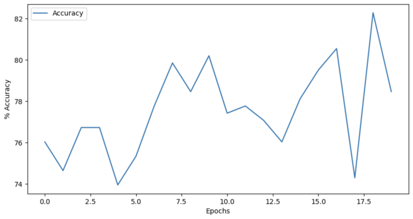


Figure 7: Humerus DenseNet

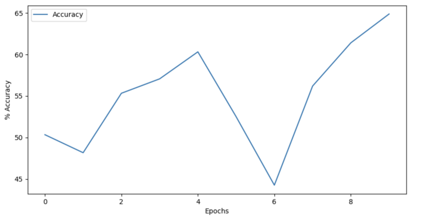


Figure 8: Forearm DenseNet

References

[1] Barua, K.; Mahmud, T.; Barua, A.; Sharmen, N.; Basnin, N.; Islam, D.; Hossain, M. S.; Andersson, K.; and Hossain, S. 2023. Explainable AI-Based Humerus Fracture Detection and Classification from X-Ray Images. In *2023 26th International Conference on Computer and Information Technology (ICCIT)*, 1–6.

[2] Huang, G.; Liu, Z.; van der Maaten, L.; and Weinberger, K. Q. 2018. Densely Connected Convolutional Networks. arXiv:1608.06993.

[3] Lindsey, R.; Daluiski, A.; Chopra, S.; and et al. 2018. Deep neural network improves fracture detection by clinicians. *Proceedings of the National Academy of Sciences*, 115(45): 11591–11596.

[4] Rajpurkar, P.; Irvin, J.; Bagul, A.; Ding, D.; Duan, T.; Mehta, H.; Yang, B.; Zhu, K.; Laird, D.; Ball, R. L.; et al. 2017. MURA: Large Dataset for Abnormality Detection in Musculoskeletal Radiographs. *arXiv preprint arXiv:1712.06957*.

[5] Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:1505.04597.

[6] Seth, P. 2022. Body$part - MURA. Kaggle.$