Bone Abnormality Detection from X-Rays using Deep Learning

MSc in Data Science / NCSR Demokritos
Deep Learning

Kastelianou Olympia
2022202004006
dit2006dsc@office365.uop.gr

[Public]

Introduction

- The main objective of this project is to build a deep learning model that given a patient study containing X-Ray images, decides if the study is normal or abnormal.

- Two different architectures used, a CNN that created from scratch and one pre-trained CNN (ResNet50).

- The MURA dataset used for the to train and evaluate of the models.

[Public]

Data and preprocessing

MURA dataset:

- 40.561 multi-view radiographic images
- 14.863 studies from 12.173 patients
- 7 study types *Elbow, Finger, Forearm, Hand, Humerus, Shoulder*, and *Wrist*
- Each study contains one or more images and is manually labeled by radiologists as either normal or abnormal.

[Public]

* *https://stanfordmlgroup.github.io/competitions/mura/*

Data and preprocessing

MURA dataset:

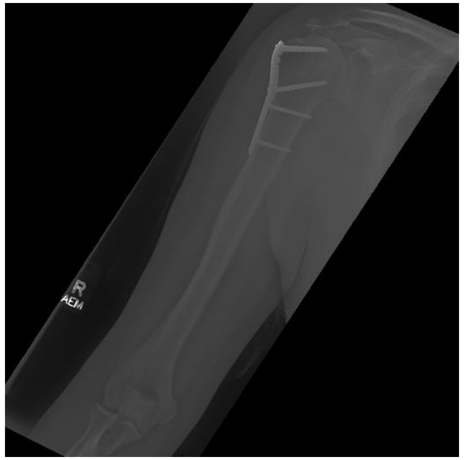
- 40.561 multi-view radiographic images
- 14.863 studies from 12.173 patients
- 7 study types *Elbow, Finger, Forearm, Hand, Humerus, Shoulder*, and *Wrist*
- Each study contains one or more images and is manually labeled by radiologists as either normal or abnormal.

Abnormal (elbow)

Abnormal (finger)

Normal (forearm)

Abnormal (humerus)



Abnormal (hand)

Normal (wrist)

[Public]

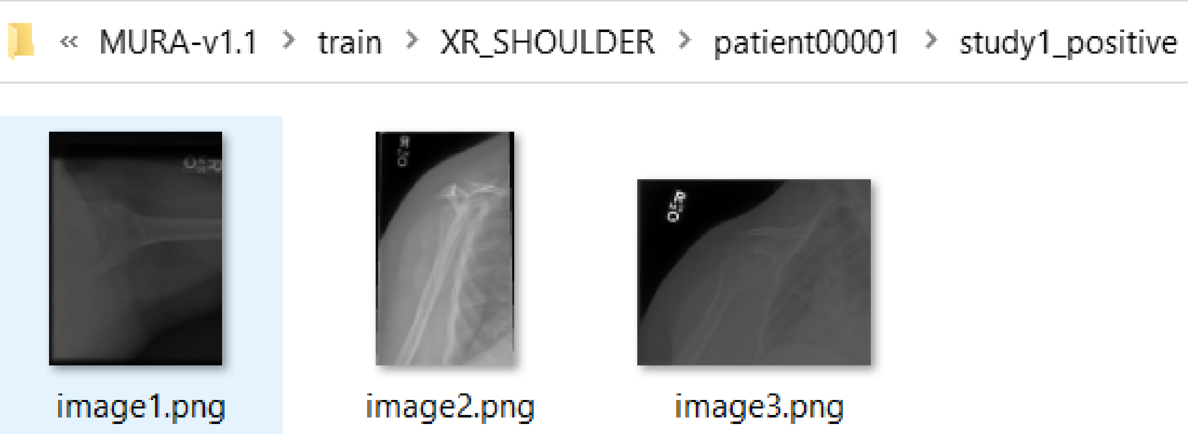
* *https://stanfordmlgroup.github.io/competitions/mura/*

MURA dataset

MURA dataset is already separated into a **training dataset** and a validation dataset.

train_image_paths.csv

```
MURA-v1.1/train/XR_SHOULDER/patient00001/study1_positive/image1.png
MURA-v1.1/train/XR_SHOULDER/patient00001/study1_positive/image2.png
MURA-v1.1/train/XR_SHOULDER/patient00001/study1_positive/image3.png
MURA-v1.1/train/XR_SHOULDER/patient00002/study1_positive/image1.png
MURA-v1.1/train/XR_SHOULDER/patient00002/study1_positive/image2.png
MURA-v1.1/train/XR_SHOULDER/patient00002/study1_positive/image3.png
MURA-v1.1/train/XR_SHOULDER/patient00003/study1_positive/image1.png
MURA-v1.1/train/XR_SHOULDER/patient00003/study1_positive/image2.png
MURA-v1.1/train/XR_SHOULDER/patient00003/study1_positive/image3.png
```



* *https://stanfordmlgroup.github.io/competitions/mura/*

MURA dataset

MURA dataset is already separated into a training dataset and a **validation dataset**. The validation dataset will be used as test dataset.

valid_image_paths.csv

```
MURA-v1.1/valid/XR_WRIST/patient11185/study1_positive/image1.png
MURA-v1.1/valid/XR_WRIST/patient11185/study1_positive/image2.png
MURA-v1.1/valid/XR_WRIST/patient11185/study1_positive/image3.png
MURA-v1.1/valid/XR_WRIST/patient11185/study1_positive/image4.png
MURA-v1.1/valid/XR_WRIST/patient11186/study1_positive/image1.png
MURA-v1.1/valid/XR_WRIST/patient11186/study1_positive/image2.png
MURA-v1.1/valid/XR_WRIST/patient11186/study2_positive/image1.png
MURA-v1.1/valid/XR_WRIST/patient11186/study2_positive/image2.png
MURA-v1.1/valid/XR_WRIST/patient11186/study2_positive/image3.png
```

« MURA-v1.1 › valid › XR_WRIST › patient11185 › study1_positive

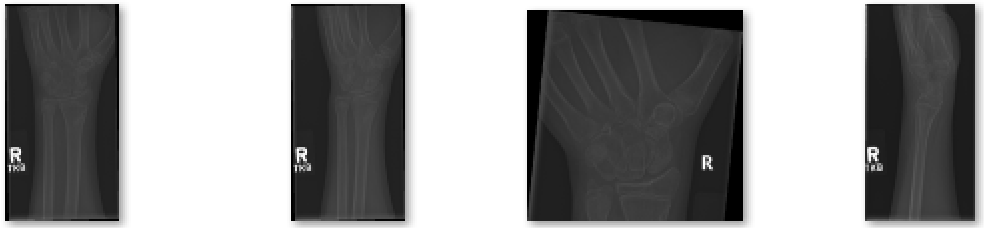


image1.png image2.png image3.png image4.png

[Public]

* *https://stanfordmlgroup.github.io/competitions/mura/*

Data and preprocessing

Data preprocessing:

- Create column “patient_study_type”
- Create column “class” (0: negative, 1: positive)
- Split the original train dataset into train (90%) and validation set (10%) based on type and class.

[Public]

Data and preprocessing

Data preprocessing:

- Create column “patient_study_type”
- Create column “class” (0: negative, 1: positive)
- Split the original train dataset into train (90%) and validation set (10%) based on type and class

Image generators:

- Two different generators for the training set were created. The first being a plain only to load the dataset as-is and a second being an augmented:
  - Random flipping the images horizontally / vertical
  - Random rotation of the images up to 5 degrees
  - Shuffling images

[Public]

Network architecture and training

Two models:

- Convolutional neural model (CNN)
- ResNet50 Pre-trained Model

Callbacks:

- **EarlyStopping**: Override implementation of keras in order to get best epoch instead of last when max epochs are reached
- **ReduceLROnPlateau**: Guarantees that the maximum accuracy will be reached

[Public]

Network architecture and training

Convolutional neural model (CNN):

- Input Layer: 224X224X3
- Conv2d Layer:
  - Filters: They grow exponentially within each layer so as to get more details from each layer
- MaxPool2d: Reduce dimensions
- Dropout: Eliminate chances of overfitting
- Flatten: So that the output layer takes a vector
- Output: Dense with activation function Sigmoid

[Public]

Network architecture and training

Convolutional neural model (CNN):

- Input Layer: 224X224X3
- Conv2d Layer:
  - Filters: They grow exponentially within each layer so as to get more details from each layer
- MaxPool2d: Reduce dimensions
- Dropout: Eliminate chances of overfitting
- Flatten: So that the output layer takes a vector
- Output: Dense with activation function Sigmoid

ResNet50 Pre-trained Model:

- Get the Resnet model with its weights, without the top Layer
- Create an MLP to replace the top layer of Resnet
- Train only the MLP for a number of epochs
- Train the whole network with a very small Learning Rate, so that pretrained weights are not vanished

[Public]

Network architecture and training

Models Compilation:

- loss: binary_crossentropy
- metrics: accuracy
- optimizer: Adam

Train parameters:

- batch_size = 32
- epochs_upper = 8 *(for MLP only)*
- epochs = 20
- epochs_trial = 1
- trials = 1
- IMG_HEIGHT = 224
- IMG_WIDTH = 224

[Public]

Hyperparameter Tuning

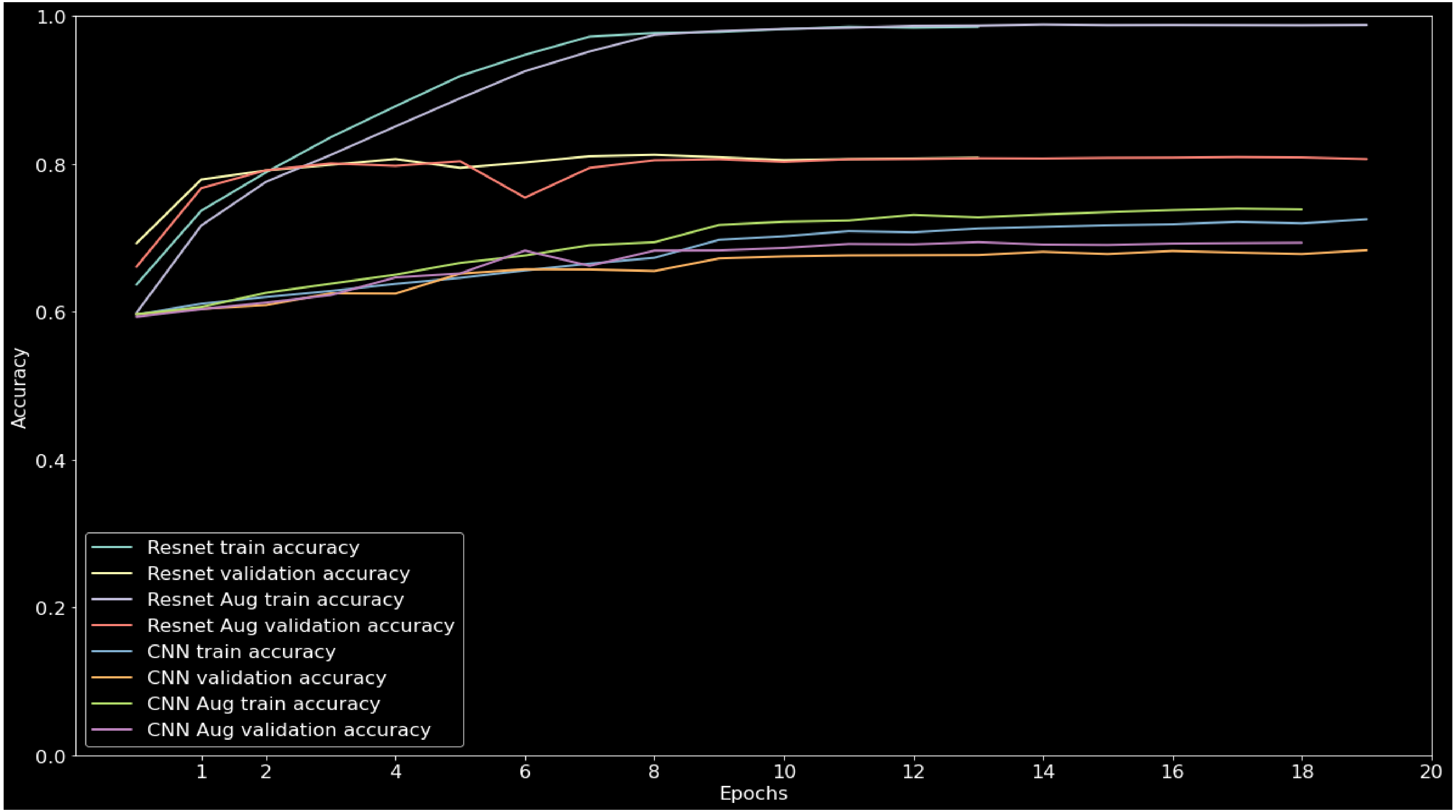
Bayesian Optimization of Keras Tuner used instead of brute forcing all possible combinations of hyperparameters.

Given more resources the search space of the hyperparameters would be the following:

- **CNN**
  - Layers:[2, 3, **4,** 5]
  - Kernel_Size:[(2, 2), (**3, 3**), (4, 4)]
  - Strides:[(**1, 1**), (2, 2), (3, 3)]
  - Dropout Rate: [0.0, 0.1, 0.2, **0.3**, 0.4]
- **MLP Layer of Resnet**
  - Layers:[2, **3**, 4, 5]
  - Units:[256, **512**, 1024]
  - Dropout Rate: [0.0, 0.1, 0.2, **0.3**, 0.4]

[Public]

Accuracy per model per set



[Public]

Results

The overall probability of abnormality for the study (patient_study_type) is calculated based on the average and the max probabilities.

Model	Validation - Accuracy (%)	Test - Accuracy (%)
CNN non-Augmented Original	68.4	68.4
CNN non-Augmented AVG	69.1	69.1
CNN non-Augmented MAX	68.5	67.0
ResNet non-Augmented Original	**81.3**	76.8
ResNet non-Augmented AVG	**81.4**	79.3
ResNet non-Augmented MAX	**81.3**	76.8
CNN Augmented Original	69.4	69.4
CNN Augmented AVG	70.4	70.4
CNN Augmented MAX	69.4	65.6
ResNet Augmented Original	81.0	**77.9**
ResNet Augmented AVG	81.1	**81.2**
ResNet Augmented MAX	81.2	**77.9**

[Public]

Thank you!

[Public]