Dataset

FractAtlas Dataset

FractureAtlas is a musculoskeletal bone fracture dataset with annotations for deep learning tasks like classification, localization, and segmentation. The dataset contains a total of 4,083 X-Ray images.
Labels for classification - Fractured and Non_fractured

Dataset Split Up
Train : Test : Val - 70 : 15 : 15

Step 1 - Baseline model using Resnet and Densenet.

Here, we take Resnet and Densenet models that have been primarily trained using the Imagenet dataset. These models are then fine tuned for our use case of detecting the fracture.

Preprocessing the images

- Resizing the input image to have a size of 256x256 pixel to maintain the aspect ratio
- Cropping the given image at the center to have a final size of 224x224 pixels.This step is typically used to ensure that the image has a standard size as used in Imagenet before being input into a neural network
- Normalizing the image tensor with the specified mean [0.485, 0.456, 0.406] and standard deviation [0.229, 0.224, 0.225] for each channel (RGB). These are the mean and standard deviation for the ImageNet dataset

These preprocessed images are then fed to the model for fine tuning.

Results obtained after fine tuning

1. Resnet - Validation accuracy of 82.54%
2. Densenet - Validation accuracy of 86.30%

As the Densenet model provided better results, as a next step we decide to perform the hyperparameter tuning.

Step 2 - Hyperparameter Tuning for Densenet.

Here, we perform hyperparameter tuning for the Densenet model.
We use the Bayesian hyperparameter tuning here, Bayesian Hyperparameter Tuning is a method for optimizing hyperparameters in machine learning models. Unlike grid search or random search, which either exhaustively or randomly try combinations of hyperparameters,

Bayesian optimization builds a probability model of the objective function and uses it to select the most promising hyperparameters to evaluate in the true objective function.

Results obtained

Iteration	lr	batch_size	epochs	optimizer	weight_decay	dropout_rate	train_accuracy	val_accuracy	train_loss
1	0.00081697	64	2	SGD	0.00027328	0.3	0.78691393	0.83660131	0.50814796
2	0.00857816	32	3	Adam	0.00260912	0.5	0.80755773	0.83496732	0.56885522
3	0.00121831	32	2	SGD	0.00253752	0.3	0.74142757	0.83496732	0.61632469
4	0.00010041	64	3	Adam	0.00098128	0.5	0.82435269	0.84803922	0.44835951
5	0.00015219	32	2	Adam	0.00049772	0.5	0.84569629	0.85620915	0.38545187
6	0.00019763	32	2	Adam	0.00045135	0.5	0.80720784	0.86111111	0.44983566
7	0.00174714	32	3	Adam	0.00405796	0.1	0.81980406	0.84150327	0.43396618
8	0.00059154	64	2	Adam	0.00016611	0.1	0.8383485	0.85784314	0.38872551
9	0.00115184	32	2	SGD	0.00012141	0.1	0.68964311	0.83496732	0.66000359
10	0.00050253	32	3	Adam	0.00014239	0.5	0.84219734	0.85130719	0.37751589
11	0.00024305	64	2	SGD	0.000756	0.3	0.82225332	0.84640523	0.67446274
12	0.00047842	32	2	Adam	0.00309074	0.5	0.83170049	0.84803922	0.41996621
13	0.00027613	32	2	SGD	0.00192454	0.3	0.84674598	0.85130719	0.64132417
14	0.00055099	32	3	SGD	0.00024376	0.3	0.81840448	0.83496732	0.65869341
15	0.00023802	64	2	SGD	0.00016591	0.5	0.84464661	0.85130719	0.66489367

After hyperparameter tuning, the best result (model parameter values which have the highest validation percentage) were obtained for the following parameters:

Best Hyperparameters -

Batch Size: 32,
Epochs: 2,
Optimizer: Adam,
Learning Rate: 0.0002
Weight Decay: 0.00045
Train Accuracy: 0.8072
Val Accuracy: 0.8611
Test Accuracy: 0.8515