Abnormality Detection in Musculoskeletal Radiographs

Jaime Cernuda Garcia CWID: 20432547

Rodrigo Aranguren Carmona CWID: 20432501

Abstract

In this project we develop a model to detect abnormalities in radiograph studies. We use a 169-layer network based on a DenseNet pretrained on ImageNet, replacing the top layer with a dense sigmoid activated layer. We achieve a 83% accuracy on the validation set. Additionally, we produce visual results with the class activation mappings for the last layer.

Problem Statement

The musculoskeletal abnormality detection task is particularly critical as more than 1.7 billion people are affected by musculoskeletal conditions worldwide. These conditions are the most common cause of severe, long-term pain and disability with 30 million emergency department visits annually and increasing[1]. Developing systems to aid doctors in performing this task with higher performance, lower error rate and with less time consumed can aid the healthcare industry to provide better patient treatment.

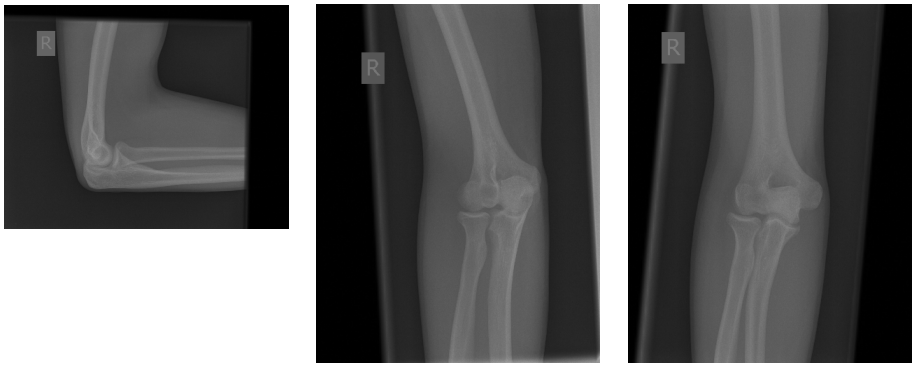
This project aims to build one possible system that performs this task with comparable performance to that shown by professionals in the industry. Applied to the are of deep learning, this will be a binary classification problem (the classes being normal or abnormal radiograph).

Table 1: Sample selection of normal and abnormal studies. Each study has one or more images of the same view.

Abnormal finger study			
Normal hand study			

Abnormal wrist study

Normal elbow study



We have to take into consideration the inherent ambiguity of this problem, as even experienced professionals often may disagree amongst themselves when analyzing a single image. Therefore, our initial objective will be to reach a comparable performance, rather than directly improving upon it.

Another side effect of the ambiguity of the problem is the difficulty to obtain properly labeled images. We will make use of one of the largest datasets available for this specific task: the MURA dataset[2]. It contains 14,863 musculoskeletal studies of upper extremities, where each study contains one or more views of the studied region and where each study has been manually labeled by radiologists as either normal or abnormal. There is a total of 40,561 images.

Following the methodology that is used by professionals, we will use several images to classify a patient. Our model will make use of the combined information provided by the images to make a decision. Mathematically speaking, we will arithmetically combine the output probabilities of the model.

Generally speaking, the nature of this problem is relatively straight-forward, as it consists of simply classifying medical images into two classes. This has extensively been done in the past with different architectures[3]–[5].

Proposed solution

We will use a 169-layer convolutional neural network to predict abnormality probabilities. The model is based on the Dense Convolutional Network architecture, described in [6]. This network is based on the concept of skip connections popularized by Highway Networks[7] and ResNet[8]. Like these, it prioritizes having a large number of layers. They argue that they can achieve higher performance compared to shallower networks. In order to not dramatically explode the number of parameters, they make use of the mentioned skip connections. DenseNet differentiates itself connecting every other layer in a feed-forward fashion, instead of only doing that in strictly successive layers. This increases the number of layer connections almost exponentially: whereas traditional convolutional networks with L layers have L connections, DenseNets has L(L+1)/2 connections.

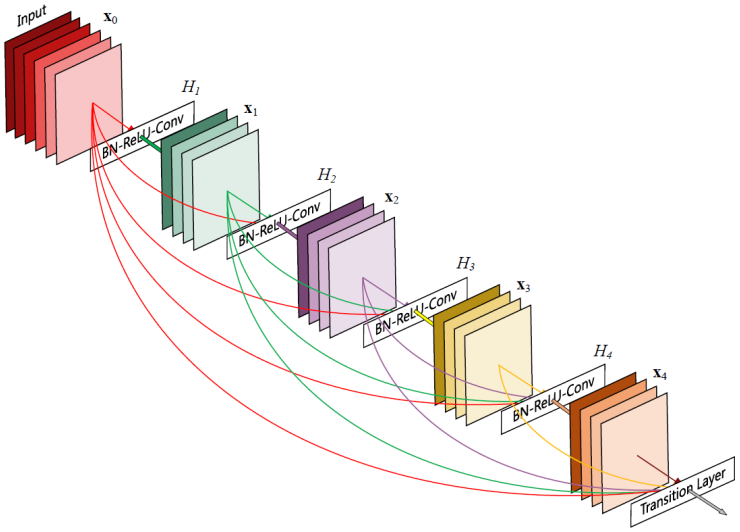


Image 1: a 5 layer block of a DenseNet. This one has a growth rate of k=4.

DenseNets allow networks to achieve a very deep architecture without suffering from vanishing gradients. This architecture strengthens feature propagation and encourages feature reuse.

This architecture is easily available with weights pretrained on the ImageNet large scale dataset for image recognition[9] directly from Keras[10].

Our model will make use of the bottom layers of the 169-layered DenseNet, with the top layer replaced by a single dense layer with sigmoid nonlinearity applied.

On top of the model design, some additional design issues need to be considered:

- The use of pre-trained weights requires us to read our grayscale images as RGB colored images. After that, we scale the images down to be 320x320 (original images may be as big as 1500x2000). To prevent the values in the network to explode, we normalize pixel values to be in the range 0-1. We also have a method to standardize the channels to have the same mean and standard deviation as the images in the ImageNet dataset.

- During training, images will be augmented using random lateral inversions and random rotations between -30 and 30 degrees.

- The instance of each label in the dataset for each section of the upper limb is not balanced, as such, the loss and accuracy calculations of the model need to take into account the different label weights, which can be dynamically provided within the data generator. Thus, the custom loss has the following equation:

  $$L(X, y) = -w_{T,1} y \log p(Y = 1|X) - w_{T,0} (1 - y) \log p(Y = 0|X)$$

  Here, $w_{T,1} = |N_T|/(|A_T| + |N_T|)$ and $w_{T,0} = |A_T|/(|A_T| + |N_T|)$ where A_T and N_T are the number of abnormal and normal images of study type T in the training set respectively.

- We want to produce class activation mappings for the last layer of the model. This way we can show how the network is giving attention to specific features or zones in an image in order to make a decision.

Development

The development was developed into two main big routes, the first one, where the training of the model was rooted in using transfer learning with a low number of epochs in each step, and a second one where the ImageNet weights were considered as weight initializers and the model was trained completely unlocked for a long number of epochs.

Due to the interconnected nature of the model used, transfer learning proved to be somewhat unsuccessful, with the model never improving upon naïve models. After many trials, the second approach was attempted.

The second approach consisted of unlocking the full network and using the ImageNet weights as initializers for the layers on the network. In this approach, we found the model very prone to overfitting, in order to solve this issue we included image argumentation into the program, where a data generator randomly decides to flip the images horizontally and to rotate the images between -30 and 30 degrees.

A small guided hyperparameter search was performed to find the ideal optimizer and the initial learning rate. Implementing a reduction of the learning rate on a plateau was also implemented, by means of keras callbacks, but it slows down the model training unnecessarily.

Once the model was trained, we implemented two additional sections. The first one calculates the different metrics for the test dataset while the second one is a prototype deployment of the model where a study is accepted and we obtain the output probability and the CAMs of the model for each image, visualization can be seen on the results section of this document. For the CAM extraction, the keras visualization toolkit was used[11].

Results

We trained the model for 100 epochs over the images belonging to a specific section of the upper limb, in our case, the wrist. The evolution of the accuracy and the loss can be seen in the following graphs:

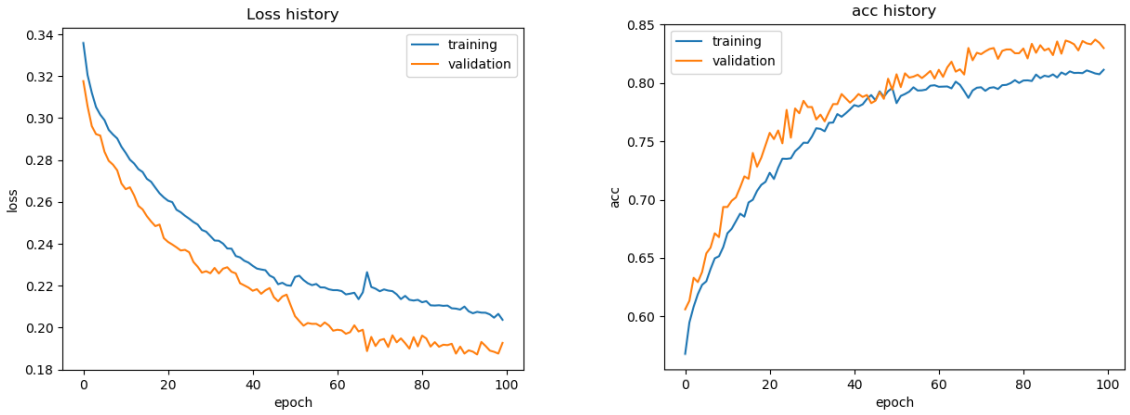


Image 2: training and validation loss and accuracy metrics.

Observing the evolution of the metrics, the model with the lowest validation loss and the highest accuracy is the model trained in epoch 91. Once selected, we run the model over the test dataset while calculating additional metrics, the results can be seen on the table below.

Table 2: validation metrics for training over 100 epochs.

Metric	**Values**
Accuracy	0.83122
Recall	0.63917
Precision	0.92537
AUC	0.88976

These metrics are relatively close to those obtained by the baseline model in [2]. However, considering the class weights are not evenly balanced, we still see room for improvement.

As an interesting experiment, we can additionally show how the network is paying attention to specific features to determine abnormality. Thus, we produce a visualization of the class activation mappings for some patients.

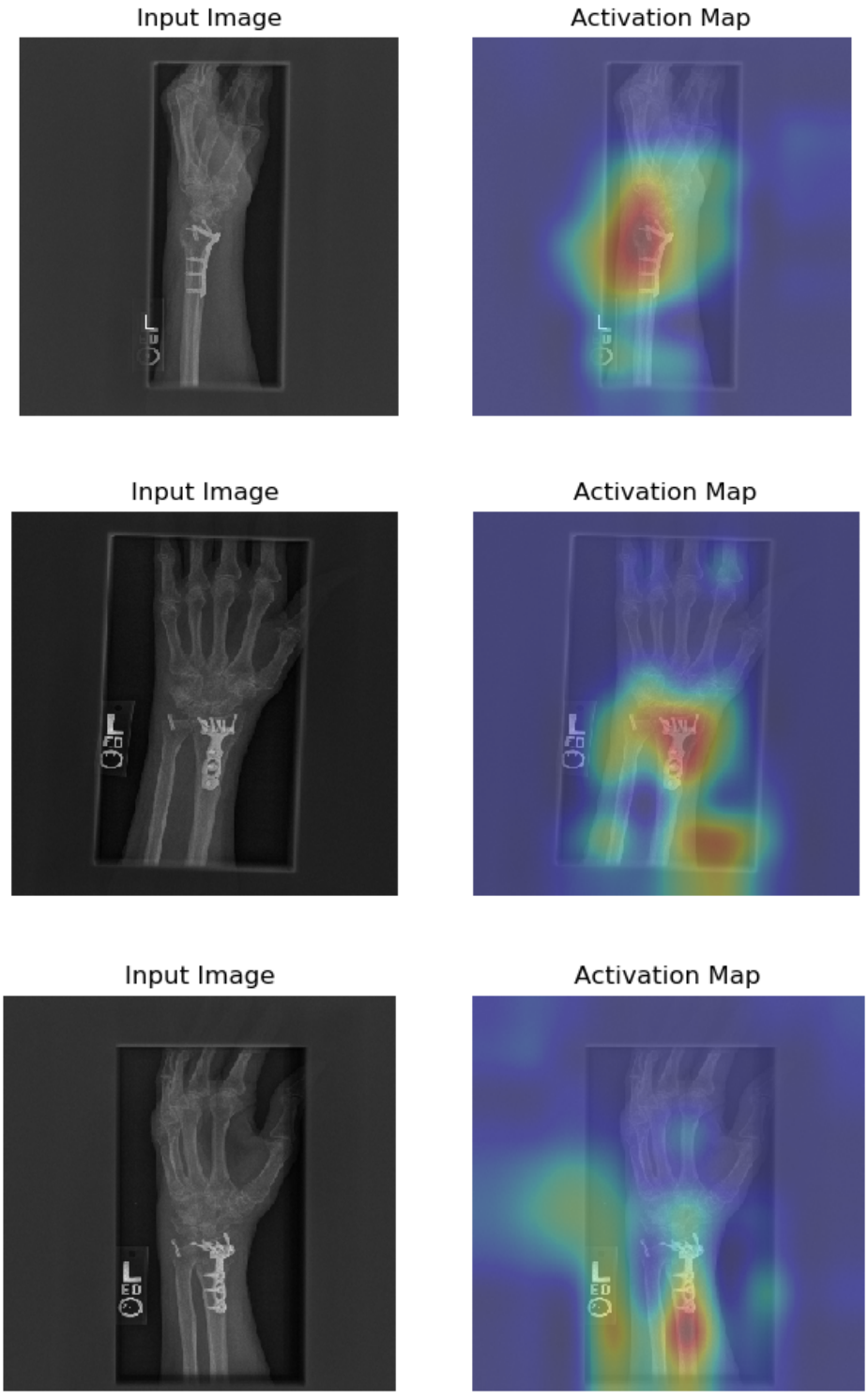


Image 3: Results for a positive study on client 11256 with a visualization of the class activation maps of the last layer of the model. The model output for the study is 0.89830595.

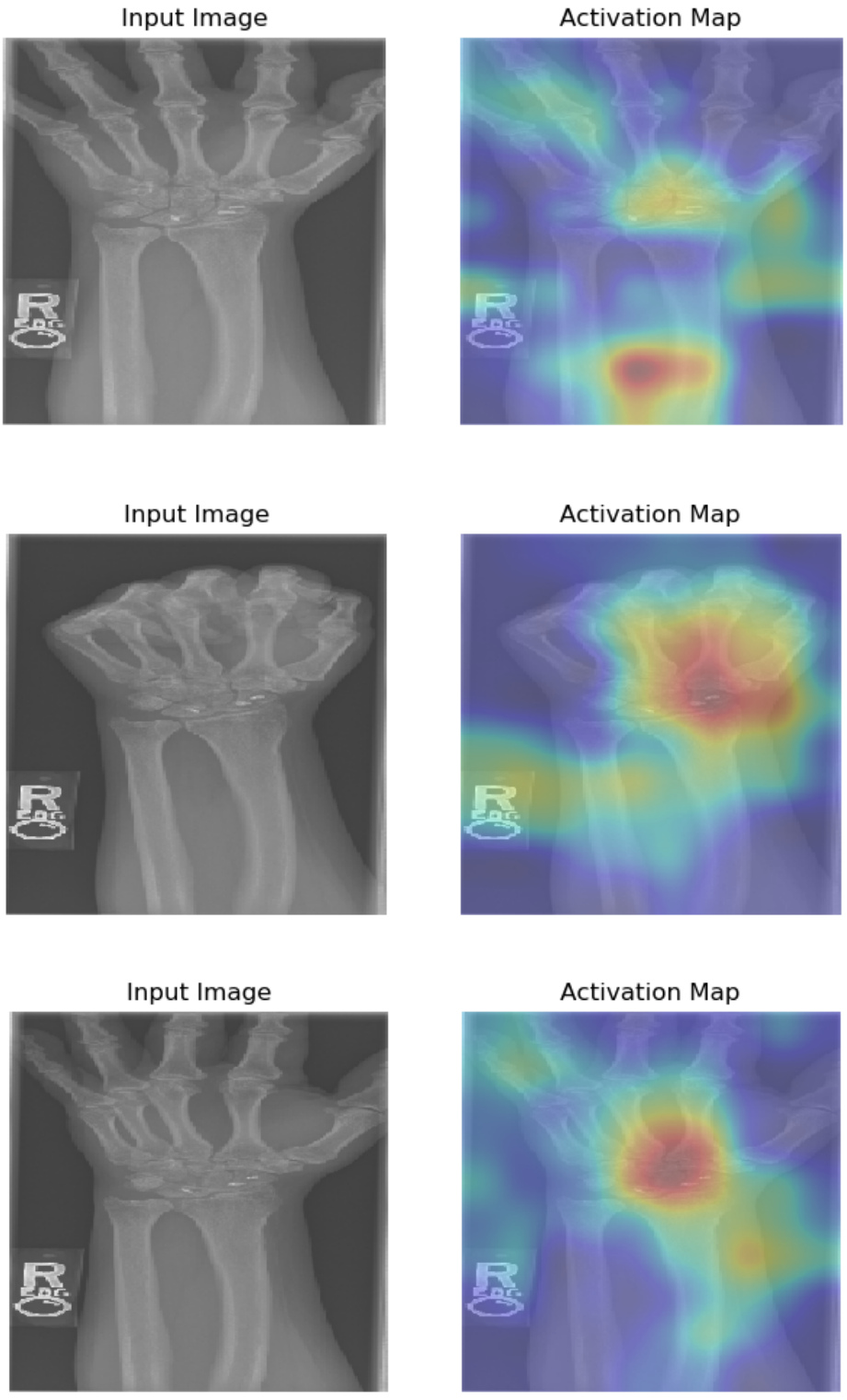


Image 4: Results for a positive study on client 11195 with a visualization of the class activation maps of the last layer of the model. The model output for the study is 0.75111933

Further Research:

There are a few aspects we have observed that could be beneficial to this project. They are left open for further work on the topic:

- Ensemble classifier. It has been proposed by [2] to ensemble the five models with the lowest validation loss or a different metric.
- Due to the area of study where we are trying to solve a problem, we could use either precision, recall or a combination of both (like the F1 metric) instead of accuracy to determine the goodness of fit for this model.
- Further training the model on all the different sections included in the MURA dataset. Even though this is relatively easy, we were very limited in time and memory:, every batch took more than an hour to run. For the limited scope of this project, this was not feasible..
- Image preprocessing: several radiograph images present a standout watermark which could affect the output probability. A possible improvement on the MURA dataset (not specific to our architecture) could consist of removing this watermark to make clean up the input to any model.
- The current model architecture uses a model trained on ImageNet. Because this is a model that uses 3 channel images (i.e. color images) while our images are grayscale, we are unnecessarily using more channels than needed. An improvement would be to convert the original colored ImageNet weights into grayscale weights. This would reduce the number of parameters significantly.

Provided program

Everything mentioned in this report is implemented and documented in the provided Python script. It makes use of Keras (TensorFlow backend) and other general use computation libraries. In order to run the program, call with the appropriate parameters:

```
python train.py
```

To review the manual of the program, call it with the `-h` argument:

```
optional arguments:
  -h, --help            show this help message and exit
  -r, --resume          Resume training from last saved model
  -s STAGE, --stage STAGE
                        Set stage of training:
                           0-train only dense layer
                           1-train dense layer with image augmentation.
                           2-train dense with augmentation and last conv block.
                           3-testing, report all metric of the test data.
                           4-evaluate a single patient, indicated with -c, plot images
                           and CAM.
  --train_path TRAIN_PATH
                        Path to train_labeled_studies.csv
  --train_images TRAIN_IMAGES
                        Path to train_image_paths.csv
  --test_path TEST_PATH
                        Path to valid_labeled_studies.csv
  --test_images TEST_IMAGES
                        Path to valid_image_paths.csv
  --model_path MODEL_PATH
                        Path to a model to resume or proceed with transfer
                        learning
  -c CLIENT, --client CLIENT
                        Client to evaluate
  -e EPOCHS, --epochs EPOCHS
                        number of epochs to train
  --section SECTION     XR_SHOU, XR_HUME, XR_FORE, XR_HAND, XR_ELBO, XR_FING,
                        XR_WRIS
  -b BATCH_SIZE, --batch_size BATCH_SIZE
                        batch size
  --print_summary       print model's summary
```

In order to train the model with the original MURA dataset, it has to be downloaded from the MURA Deep Learning Competition (freely available after signing a Use Agreement).[1]

[1] Avaliable at https://stanfordmlgroup.github.io/competitions/mura/. Direct link to the dataset is forbidden.

Bibliography and references

[1] "BMUS: The Burden of Musculoskeletal Diseases in the United States," *BMUS: The Burden of Musculoskeletal Diseases in the United States*. [Online]. Available: https://www.boneandjointburden.org/. [Accessed: 23-Apr-2019].
[2] P. Rajpurkar *et al.*, "MURA: Large Dataset for Abnormality Detection in Musculoskeletal Radiographs," *ArXiv171206957 Phys.*, Dec. 2017.
[3] P. Rajpurkar *et al.*, "CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning," *ArXiv171105225 Cs Stat*, Nov. 2017.
[4] P. Rajpurkar, A. Y. Hannun, M. Haghpanahi, C. Bourn, and A. Y. Ng, "Cardiologist-Level Arrhythmia Detection with Convolutional Neural Networks," *ArXiv170701836 Cs*, Jul. 2017.
[5] G. Erion, H. Chen, S. M. Lundberg, and S.-I. Lee, "Anesthesiologist-level forecasting of hypoxemia with only SpO2 data using deep learning," *ArXiv171200563 Cs Stat*, Dec. 2017.
[6] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *ArXiv160806993 Cs*, Aug. 2016.
[7] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway Networks," *ArXiv150500387 Cs*, May 2015.
[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *ArXiv151203385 Cs*, Dec. 2015.
[9] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *ArXiv14090575 Cs*, Sep. 2014.
[10] "Applications - Keras Documentation." [Online]. Available: https://keras.io/applications/. [Accessed: 23-Apr-2019].
[11] keras Raghavendra Kotikalapudi, *keras-vis Neural network visualization toolkit for keras*. 2017.