Detecting Musculoskeletal Conditions

Gökçe Şengün [1] **Hüseyincan Kaynak** [1] **Utku İpek** [1]

Abstract

Nowadays, machine learning can be used to solve many types of problems. In this paper, we introduce our machine learning models we have developed for BBM 406: Fundamentals of Machine Learning Course Project with the theme "Machine Learning for Good", to solve a specific kind of problem we have found on Stanford ML Group website. The problem is to detect whether a given bone X-ray has a musculoskeletal condition, which are abnormalities in the muscle-bone system. The dataset we have used for the project is MURA, a dataset that contains bone X-rays of seven parts of the body, which are shoulder, humerus, elbow, forearm, wrist, hand, and finger(1). At first, we have only focused on one body part which is the wrist and developed our models on that data. When we are done improving our models, we have trained our best model with the hand data. One of our models is a Convolutional Neural Network, and the other is Densely Connected Convolutional Network(3). We hope that we will improve our results in the future since our models tend to be improved.

1. Introduction

Musculoskeletal Condition is a disorder about your bones and muscles. This condition also is known as a musculoskeletal disorder. It is an issue at the human's muscle system and between the human's bones system. Many people have that disorder, actually beyond that many more than 1.7 billion people around the world having this disorder(1). Many of our daily activities are cause that disorder. We don't care about our sitting position or most of the jobs are done from sitting on a desk and we don't stretch and walk at least every hour. These ordinary behaviors could cause this disorder. Symptoms are severe pain in your bones or even some disabilities. People ignore these symptoms some of them not paying attention and some of them are believing this is not a critical issue. Even if they will go to the hospital, this disorder is not easy to notice. Many health care centers don't have enough experienced radiologist that figure out this condition from an X-Ray image.

Our main motivation is that there is a disorder and people don't pay attention, even if they go to the hospital for that complaint, the world's average radiologists couldn't notice this disorder. To solve this problem is our main motivation. Further, this is a moral situation for us that has the capability of helping people who need help with this project.

To achieve our goals, we used MURA Dataset which is a very great dataset. It contains 40,561 images. In this dataset there are X-Ray images from 7 different body parts, these are wrist, shoulder, hand, finger, forearm, humerus and elbow(1). While we were researching we found many related works. We saw that almost every work used this dataset, it allowed us to learn many methods and achieve more accurate results. Many people used DenseNet for this case. We use DenseNet169 which is a type of DenseNet that also related work uses(4). DenseNet169 has 169 layers and more than 12 million parameters. In our implementation, we have used Keras, which is a library that makes it easy to implement deep learning algorithms (5) and it has the DenseNet169 application(6). It is by default for ImageNet, get input size as 224x224 and we configure it our images according to it. We change the parameter for our project. In our project, it accepts 224x224x3 size inputs and set classes to 2 due to we are classified concerning positive or negative. We added a dense layer to the end of the DenseNet for predictions. Since we used Google Colab we set batch size 8 because Colab has some limitations on RAM and GPU. We also implement a Convolutional Neural Network model. It has 12 layers. It consists of convolutional, pooling, dense and flattens layers. For this model, we set the batch size to 64 and for input size we again used 224x224x3. We got results for two models on different parts of the body and we observe that result and interpret these results.

[1]Hacettepe University, Department of Computer Engineering. Correspondence to: Gokce Sengun <gokcesengun@hacettepe.edu.tr>, Hüseyincan Kaynak <huseyincankaynak@hacettepe.edu.tr>, Utku İpek <utkuipek@hacettepe.edu.tr>.

BBM406 Fundamentals of Machine Learning, Fall 2019.

2. Related Work

While researching our project, we found that there are many studies on the detection of abnormalities in muscles and bones from X-ray images. The reason is that there was a competition that was held on this subject. The most popular one is the competition with the Mura dataset. Andrew Ng, Aarti Bagul, Robyn L. Ball, Brandon Yang, Kaylie Zhu, Matthew P. Lungren, Dillon Laird, Curtis Langlotz, such as many developers have provided great benefits to this competition(4).

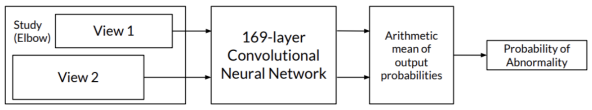

Figure 1. Shows the architecture that is used in the related work (taken from https://stanfordmlgroup.github.io/competitions/mura/).

In the data set consisted of 40,561 pictures of 12,173 patients between 2001-2012, it is primarily labeled manually by successful radiologists in their fields. There are 2 labels that an X-ray radiograph can have, positive or negative. Abnormalities in the bones and muscles can be illustrated as follows: long and constantly increasing pain, fractures, and cracks, etc. Mura dataset contains information about seven separate sections of the body: shoulder, humerus, elbow, forearm, wrist, hand, and finger(1). We also evaluated the accuracy of the model manually according to the results obtained from radiologists. As modeled in the illustration, the model examines at least one image for a parent study. It is then inserted into the CNN model consisting of 169 layers(4). It is aimed to better educate the model by taking the average of the probability of output. Wrist, hand and finger model results can compete with successful radiologist results. The data set in this research is shared publicly for the promotion of the studies in this field.

	Radiologist 1	Radiologist 2	Radiologist 3	Model
Elbow	0.850 (0.830, 0.871)	0.710 (0.674, 0.745)	0.719 (0.685, 0.752)	0.710 (0.674, 0.745)
Finger	0.304 (0.249, 0.358)	0.403 (0.339, 0.467)	0.410 (0.358, 0.463)	0.389 (0.332, 0.446)
Forearm	0.796 (0.772, 0.821)	0.802 (0.779, 0.825)	0.798 (0.774, 0.822)	0.737 (0.707, 0.766)
Hand	0.661 (0.623, 0.698)	0.927 (0.917, 0.937)	0.789 (0.762, 0.815)	0.851 (0.830, 0.871)
Humerus	0.867 (0.850, 0.883)	0.733 (0.703, 0.764)	0.933 (0.925, 0.942)	0.600 (0.558, 0.642)
Shoulder	0.864 (0.847, 0.881)	0.791 (0.765, 0.816)	0.864 (0.847, 0.881)	0.729 (0.697, 0.760)
Wrist	0.791 (0.766, 0.817)	0.931 (0.922, 0.940)	0.931 (0.922, 0.940)	0.931 (0.922, 0.940)
Overall	0.731 (0.726, 0.735)	0.763 (0.759, 0.767)	0.778 (0.774, 0.782)	0.705 (0.700, 0.710)

Figure 2. Shows the radiologist performance compared to related work (taken from https://stanfordmlgroup.github.io/competitions/mura/).

Only the performance of the model is not evaluated in the test set part of the work in the related work. Also, this performance is compared with the performance of radiologists. According to the estimation results of each radiologist for each part of the body, two performance results are obtained. The average of the two results for each section performs a single site for each radiologist. This result is compared with the performance of the model. According to the performance obtained from radiologists, we found that they were difficult to detect in the wrist and humerus regions. For these two, we have seen worse results than the model. The best performance is in the hand area. Although the model has had difficulty in detecting abnormalities in the fingers, its performance seems to be good enough for other parts. It performs approximately in the 0.80s(4). For musculoskeletal X-ray chart results, the detection of abnormalities is very important. But before starting this project, it should be well established which anomalies will be worked on. With these determinations, we can reach the treatment time more quickly and obtain more reliable results. This related work is aware of this and work is done according to it. The use of real radiologists and the desire to develop the model within the competition is a testament to the importance given to these studies(4).

3. The Approach

Since the problem we've tried to solve is an image classification problem, firstly, our solution approach was to build a Convolutional Neural Network (CNN) model. CNN architecture contains convolutional layers and pooling layers unlike an ANN (Artificial Neural Network), in this way, it preserves the spatial structure of the given images and it can return better results. Secondly, to improve our accuracy results, we have decided to use another architecture which is a different type of convolutional network named Densely Connected Convolutional Network (DenseNet)(3). This architecture provides to make CNN's train process easy and get better accuracy results from it since there are connections between each layer.

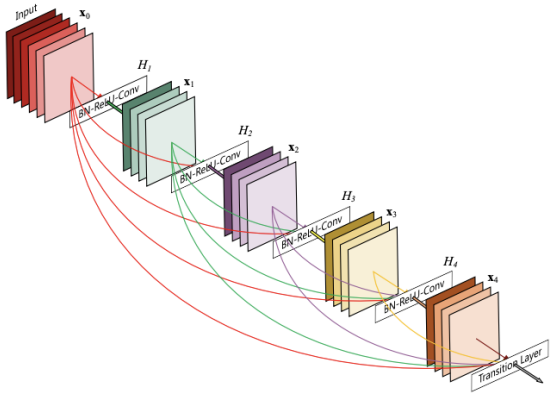

Figure 3. An example of DenseNet figure (taken from https://pytorch.org/hub/pytorch_vision_densenet/).

It is mentioned in the introduction section of the paper that the dataset we have used contains seven parts of the body. At the beginning of the project, we had planned to focus on only one part which is the wrist but then, at the end of the project, we have also tried the hand part in our DenseNet model. The reason we have chosen the wrist part is that it is the part that contains the most data among the other parts. The choice of the hand part was only to see if we train a new model with the same architecture of the model that we have trained on the wrist part on the hand images, can we get similar accuracy results. So, that means we have three models in total, one CNN and two DenseNet models. Our implementation details of CNN and DenseNet models can be found in the following subsections but before that, the next subsection gives the details of how we have made the augmentations on the MURA dataset.

3.1. Data Augmentation

First of all, the wrist part of the MURA dataset contains 9752 training data and 659 validation data. Since this was a competition, the MURA dataset does not include a test set inside of it. Our solution to that is to use the validation set as our test set and split the training set into a training set and a validation set. After we have applied this to our dataset, the data distribution for the wrist part had become 9089 training images, 663 validation images, and 659 test images. We have also resized the images as 224x224 and normalized the pixel values. After we have organized the dataset which we will use to train our CNN and DenseNet models, we have made some augmentations on the Data to expand our training data and improve our accuracy results. The first augmentation we applied to our training data is to add a 30 degrees rotation range. Then, we have applied a width shift and a height shift with ranges of 0.2 percentages of the image's height and width. Also, we have applied shearing and zooming augmentations with ranges of 0.2 percentages again. Finally, in addition to those augmentations, we have allowed the horizontal flip on the images(10)(11). For the hand part, we have applied the same augmentations we have used for the wrist part. And the numbers for the hand part are 5167 training images, 376 validation images, and 460 test images.

3.2. CNN Model

Our convolutional neural network architecture contains the following layers; three convolutional layers, three max-pooling layers, two fully-connected layers, and an output layer. In the process of creating our Convolutional Neural Network architecture, firstly, we have added a convolutional layer that has 32 feature detectors with a size of 3x3. Then, a max-pooling layer was added with a pool size of 2x2. The next convolutional layer we have added has 64 feature detectors with the same size as the first convolutional layer. Again, a max-pooling layer was added with the same pool size. Finally, a convolutional layer that has 128 feature detectors with a size of 3x3, and a max-pooling layer with a pool size of 2x2 were added again. When we are done adding convolutional layers and pooling layers, we have applied the flattening operation. After the flattening, two fully-connected layers were added with 64 nodes and 128 nodes respectively. To prevent overfitting, we have used the dropout technique by adding two dropout layers between fully-connected layers and output layer(7). The output layer we have added contains 2 nodes. That layer is for predictions.

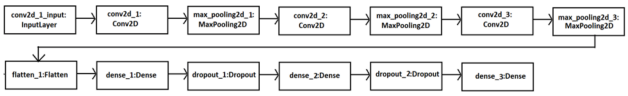

Figure 4. Our CNN architecture.

We have used ReLU between layers and Softmax in the output layer as activation functions. The loss function we have used is binary cross-entropy loss function and as an optimizer, we have used Adam with its default parameters. The results can be found in the Experimental Results section.

3.3. DenseNet Models

The DenseNet architecture we have implemented contains 169 layers. In addition to that architecture, we have added an output layer for predictions. An important point here is that in our DenseNet architecture, we have used average pooling and we have used the weights that are pre-trained on ImageNet(2), which is a large image database.

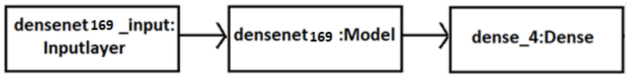

Figure 5. Our DenseNet architecture.

Like the CNN model, we have used softmax as an activation function in the output layer. The loss function we have used for the DenseNet model is again binary cross-entropy loss function. As an optimizer, we have used Adam but this time we have initialized the learning rate to 0.0001 different from our CNN model. An important note here is that we have reduced the learning rate by a factor of 10 when the validation loss has stopped improving for 5 epochs. Then, to prevent overfitting, we have saved only the model with the best validation loss while we were training the model.

4. Experimental Results

In this section, some figures show the loss and accuracy values of our models for training and validation data, and

also show the prediction results and confusion matrices with and without normalization for the test data. Besides that, we have visualized the parameters of our DenseNet model which is trained on the wrist data. The results can be found in the following subsections.

4.1. CNN Results

As seen in the loss and accuracy plots in Figure 6, we have trained our CNN model for 100 epochs. The value of train loss and validation loss have reduced by approximately 0.2 during the training process. At the end of the training, we got 0.77 train accuracy and 0.72 validation accuracy. When we have tested our model on the test set, we got 0.7 test accuracy. Also, Figure 7 shows the confusion matrices of our model.

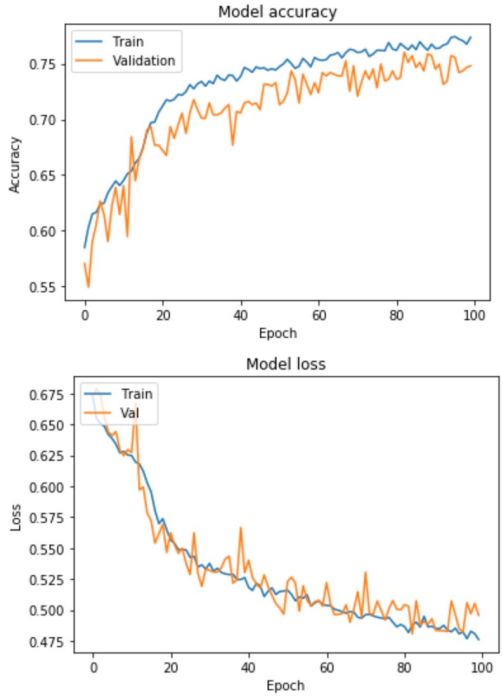

Figure 6. Loss and accuracy values of our CNN model that is trained on the wrist data.

As seen in those confusion matrices, our convolutional neural network model predicted well on the ”negative” labeled images. If we look at the model’s prediction rate for ”positive” labeled images, we see that the model did not predict well on those images.

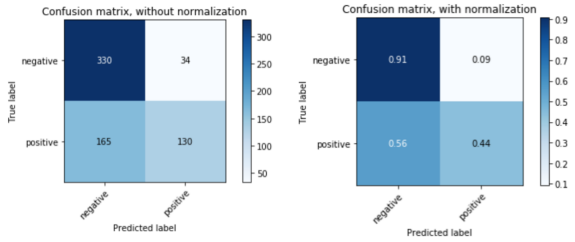

Figure 7. Confusion matrices of our CNN model with and without normalization.

4.2. DenseNet Results

We have trained both DenseNet models for 25 epochs. As seen in Figure 8, the model which was trained on the wrist part, starts to overfit the data after some point. But before overfitting, we have saved the model with the lowest validation loss value. The same case is valid for the hand part as seen in Figure 10. We have also saved that model before overfitting. For the wrist part, we got 0.84 test accuracy and confusion matrices for that model shown in Figure 9. For the hand part, we got 0.78 test accuracy and that model’s confusion matrices are shown in Figure 11.

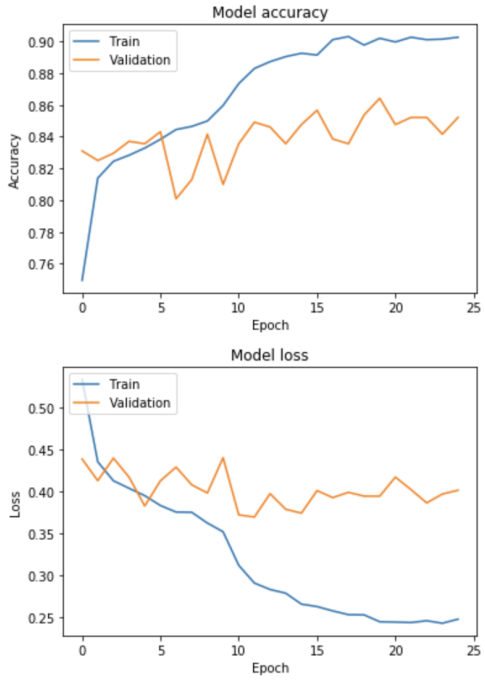

Figure 8. Loss and accuracy values of our DenseNet model that is trained on the wrist data.

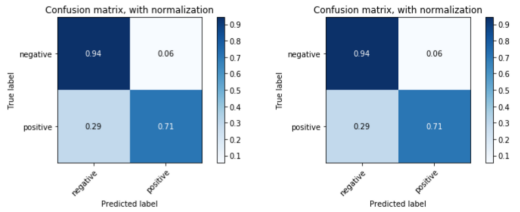

Figure 9. Loss and accuracy values of our DenseNet model that is trained on the wrist data.

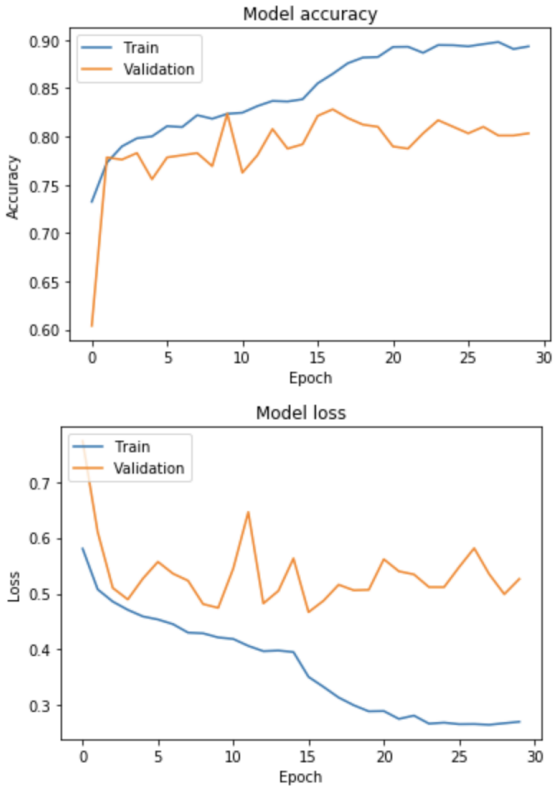

Figure 10. Loss and accuracy values of our DenseNet model that is trained on the hand data.

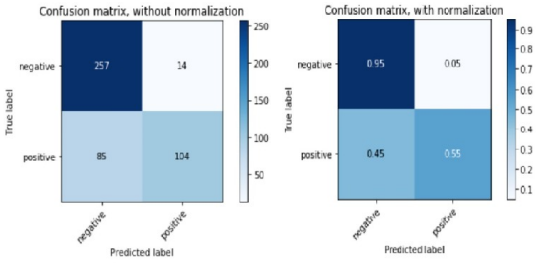

Figure 11. Loss and accuracy values of our DenseNet model that is trained on the hand data.

4.3. Visualizing The Parameters

On the images, to see where the parameters of the model are focused on, we have tried to visualize the parameters of the model by using Class Activation Map technique(8)(9). If the model predicts an X-ray as a positive (abnormal), visualized parameters can show where the abnormality is. Figure 12 shows that the negative (normal) X-ray images and their class activation maps where Figure 13 and Figure 14 show positive X-ray images and their class activation maps.

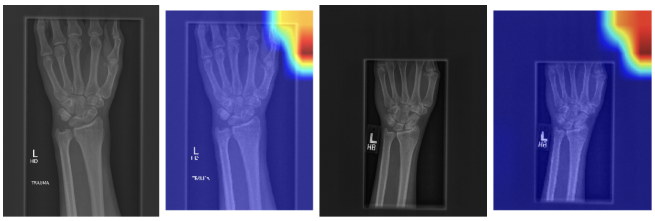

Figure 12. Class Activation Maps for two negative(normal) X-rays.

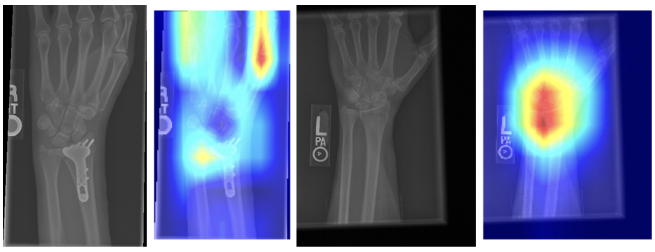

Figure 13. Class Activation Maps for two positive(abnormal) X-rays.

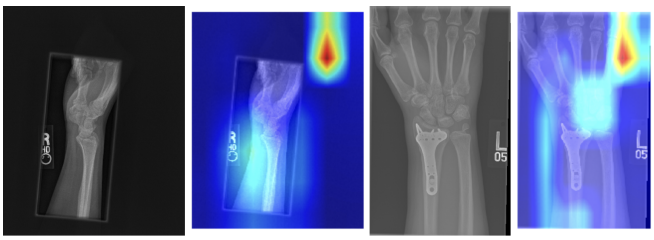

Figure 14. Class Activation Maps for two positive(abnormal) X-rays.

5. Conclusions

In this project, we have used different architectures for our models and also we've used different methods to improve our models. We have seen how CNN and DenseNet architectures work and got a chance to compare these two different architectures. Of course, the main purpose of the selection of these architectures was that they are successful in the image classification problems and they can handle

many parameters during the training process. The reason for choosing the wrist part from the dataset is based on the number of images like it is stated in the Approach section. At the beginning of the project, while we were trying to decide which parts are more advantageous, we have decided by trying the parts we have separated in our data set one by one. Then, as we developed our CNN and DenseNet models, we saw that trying out all the parts in the data set became problematic in terms of time so we have decided to experiment with the wrist part only. Table 1 shows the accuracies of our best models that are trained on the wrist data. As seen in the table, our DenseNet model has given much better results compared to our CNN model. But, if we compared our current CNN model with our first CNN model, we can say that it tends to be improved. Still, our first goal is to improve our results on the DenseNet model.

Table 1. Classification accuracies of our best CNN and DenseNet models on the wrist data.

MODEL	TRAIN	VALIDATION	TEST
CNN	75%	72%	70%
DENSENET	87%	84%	84%

When we have done training our models on the wrist part, we wanted to try one more body part on our DenseNet architecture. We have trained the DenseNet model with the hand data and it has given 78% test accuracy.

After we've trained all of our models and got all the results, we wanted to see how the best model of ours interprets the X-ray images. With the Class Activation Map tecnique(8), we have observed that which parts of the images are more focused on after the prediction process of the model. In Figure 12, we can see that if the DenseNet model predicts an image as negative (normal), there are no parameters on the wrist region. All of them focus on the right top corner of the image. And if the model predicts an image as positive(abnormal), on the wrist region, some parameters can be seen (Figure 13 and Figure 14). This may show where the abnormalities are. An important note here is that on those figures, we have only used the test images that the model's prediction is true.

If we compare our DenseNet model with the related work in the Related Work section, we can say that our results are not bad. Also, in Figure 2, we can see that even the best radiologist made mistakes while classifying the X-ray images. Of course, our models have some weaknesses. They are not that good at predicting the positive X-ray images compared to negative X-ray images. If we had a test set, we would use our whole training set to train the model, so, probably our accuracies would be higher.

Finally, the theme "Machine Learning for Good" has made us develop this project. As future work, our priority is to improve our accuracies on the wrist part. Then we are planning to train our models on the other parts of the MURA dataset. After that, with our models with the best accuracies, we are also planning to create a website for this project. This may help many people around the world who suffer from these musculoskeletal conditions.

References

[1] MURA Dataset: Towards Radiologist-Level Abnormality Detection in Musculoskeletal Radiographs https://stanfordmlgroup.github.io/competitions/mura/

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei. *ImageNet: A Large-Scale Hierarchical Image Database*. Dept. of Computer Science, Princeton University, USA.

[3] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger. *Densely Connected Convolutional Networks*. 2018.

[4] Pranav Rajpurkar, Jeremy Irvin, Aarti Bagul, Daisy Ding, Tony Duan, Hershel Mehta, Brandon Yang, Kaylie Zhu, Dillon Laird, Robyn L. Ball, Curtis Langlotz, Katie Shpanskaya, Matthew P. Lungren, Andrew Y. Ng. *MURA: Large Dataset for Abnormality Detection in Musculoskeletal Radiographs*. 22 May, 2018.

[5] Keras: The Python Deep Learning Library https://keras.io/

[6] Applications: Keras Documentation https://keras.io/applications/#densenet

[7] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. Department of Computer Science, University of Toronto Journal of Machine Learning Research 15 (2014) 1929-1958

[8] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba. *Learning Deep Features for Discriminative Localization*. Computer Science and Artificial Intelligence Laboratory, MIT, 2015

[9] Keras Class Activation Map https://github.com/nickbiso/Keras-Class-Activation-Map

[10] Image Preprocessing: Keras Documentation https://keras.io/preprocessing/image/

[11] Jason Brownlee: Image Augmentation for Deep Learning With Keras
https://machinelearningmastery.com/image-augmentation-deep-learning-keras/