Noroff University College

Machine Learning

UC3MAL101

Bone X-ray Classification

Using Deep Learning

Author
Lewi Lie Uberg

Professor
Dr. Isah A. Lawal

13th October 2020

Contents

Abstract

Musculoskeletal conditions is one of the most common causes of long term disability affecting 1.7 billion people worldwide, annually leading to leading to 30 million emergency department visits. This paper addresses the problem of automating classifying musculoskeletal radiograph images in remote locations without trained radiologists' assistance. The classification process can be automated by training a convolutional neural network (CNN) on the MURA data set. The model proposed in this paper achieves an accuracy of .70 and an F1 score of .63.

1 Introduction

This paper aims to design and implement a deep learning model, trained on the musculoskeletal radiograph images to detect abnormalities, thereby aiding the advancement of automated medical imaging systems providing healthcare access in remote locations with restricted access to trained radiologists.

2 Modeling

2.1 Justification of layer elements

The convolutional neural network (CNN) is a concept introduced by Fukushima (1980), later greatly improved by Lecun et al. (1998) that has significantly impacted computer vision. A CNN is used for pattern detection, where one or more hidden convolution layers uses filters to convolve or scan over an input matrix, such as a binary image. These filters closely resemble neurons in a dense layer, where a filter is learned to detect a specific patter such as an edge or circle; adding more filters to a convolutional layer will enable more features to be learned. The filter size is the size of the matrix convolving over the image matrix, and the stride is the number of pixel shifts over the input matrix. For each stride a matrix multiplication is performed on the image and filter matrix, which results in a feature map as the output. When the filter does not fit the image, two options are available—either dropping the part of the image matrix that does not fit the filter, which is called valid padding, or add zeros to the image matrix' edges, enabling the filter matrix to fit the image matrix entirely, this is called zero-padding. The most common activation function for a convolutional layer is the Rectified Linear Unit (ReLU) function. ReLU is an activation function with low computational cost since it is almost a linear function. Transforming the input to the maximum of zero or the input value itself makes it converge fast, meaning that the positive linear slope does not saturate or plateau when the input gets large. Unlike sigmoid or than, ReLU does not have a vanishing gradient problem. To reduce the dimensionality of a feature map, that is, the number of tunable parameters, spatial pooling can be applied, which is called subsampling or downsampling. While there are different types of spatial pooling, Max-pooling is often used in CNN's. A Max-pooling layer operates much like convolutional layers, it also uses filters and stride, but it takes the maximum value in its filter matrix as the output value. A Max-pooling layer with an input matrix of 8x8 with a filter size of 2x2 would have an output of 4x4 containing the larges value for each region. This downsampling will decrease the computational cost for the following layers in the network. This concludes the feature learning part of the CNN and commences with the feature learning part. A convolutional or max-pooling layer outputs a matrix; however, a fully-connected layer only accepts vectors. Therefore, a flattening layer is added to reduce the last convolutional layer's output dimensionality to shape (-1, 1), or transform the matrix into a vector. Adding a fully-connected layer can be a computationally cheap way of learning non-linear combinations of higher-level features represented by the convolutional or max-pooling layer's output. Hidden fully-connected layers also called dense layers in a CNN, often use ReLU as their activation function. The final layer, the output layer, is a fully-connected layer with the same number of neurons as classes to be classified. The output layers' activation function is dependent on the loss function. Using a single neuron sigmoid activated dense layer for the network's output, compiled with binary cross-entropy as the loss function would yield the same result as using two softmax activated neurons in a network using categorical cross-entropy as the loss function; in other words, a binary classification.

2.2 Model architecture

The proposed CNN model is designed to accept N amount of 128x128 image matrices with 1 color channels; it consists of three convolutional layers of 32, 64, and 64 filters, each of filter size 3x3, with zero-padding, and ReLU as the activation function. The first convolutional layer is followed by a max-pooling layer of pool size 4x4, and the last two convolutional layers have a pool size of 2x2, each followed by a dropout layer with a dropout rate of 0.15. A flattening layer is added to transform matrix output to vector inputs to be accepted by the first dense layer, which is comprised of 512 ReLU activated neurons, followed by a dropout layer with a 0.5 dropout rate. The last hidden layer is a dense layer of 256 ReLU activated neurons. The model's final layer, its output, is a 2 neuron softmax activated dense layer allowing for binary classification. The general architecture of this model is shown in figure 1.

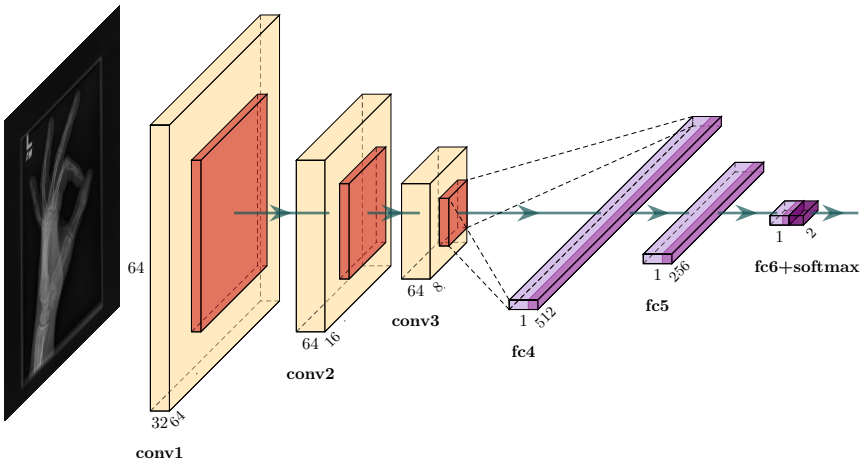
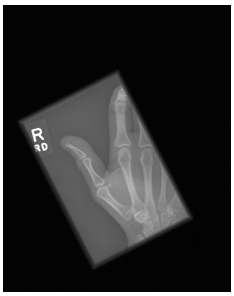


Figure 1: CNN Architecture

3 Dataset

The dataset used to train the CNN model is the publicly available Large Dataset for Abnormality Detection in Musculoskeletal Radiographs, widely known as MURA (Rajpurkar et al., 2018). The MURA dataset wast the basis for a Deep Learning competition hosted by Stanford, which expected the participants to detect bone abnormalities. MURA is a dataset of musculoskeletal radiographs consisting of 14,863 studies collected from 12,173 patients, with 40,561 multi-view radiographic images, meaning that one patient can have multiple images used for diagnosis, and seven different categories elbow, finger, forearm, hand, humerus, shoulder, and wrist. Between 2001 and 2012, board-certified radiologists from the Stanford Hospital has manually labeled each study during clinical radiographic interpretation. The dataset images vary in aspect ration and resolution, which can be beneficial during neural network training. The dataset is split into training (11,184 patients, 13,457 studies, 36,808 images), validation (783 patients, 1,199 studies, 3,197 images), and test (206 patients, 207 studies, 556 images), and there is no overlap in patients between any of the sets. The test set is not available to the general public; therefore, in preparation for training the model, the training set is further split into a new train 80% and a validation set 20%, keeping the test set for evaluation after model training. The reasoning for this decision is to be able to present the models' generalization ability and accuracy. Figure 2 shows example images of a normal figure 2a and an abnormal figure 2b diagnosis.



(a) Normal

(b) Abnormal

Figure 2: MURA Image example.

4 Experimental setup

For model development and training of the model, Google's open-source python library Keras is used as an interface to the machine learning library TensorFlow. A sequential model is defined using the Keras API with the layered architecture described in the Modeling section. The model is compiled with the stochastic gradient descent variant Adam, an adaptive learning rate optimization algorithm specifically designed for deep neural network training (Kingma and Ba, 2017). Categorical cross-entropy is used as the loss function, using accuracy as the evaluation metric. The model training is set to feed batches of 32 image arrays for 50 epochs, with an initial learning rate of .001. However, to find the optimal learning rate, the ReduceLROnPlateau method reduces the learning rate with a factor of .5 if val_loss does not decrease for two epochs. The EarlyStopping method is also implemented to keep from running if val_loss does not decrease for five epochs.

5 Discussion

Overfitting is a common problem in deep neural networks. To combat this problem, several measures can be taken. The first choice is to increase the amount of training data. If this is not possible, data augmentation like mirroring, cropping, rotating, or embossing can be performed on the available data to provide additional training scores. However, this method did not increase the accuracy of this model. In the experimental stage of model development batch normalization, as well as L1, and L2 regularization, were implemented without improving performance. Yet, regularization by adding dropout proved to be significant in decreasing overfitting. The next challenge was the model would seem to settle to local minima, and essentially stop training. To rectify this, the Nadam optimizer was implemented, which is the Adam optimizer with Nesterov momentum. Many decay rates were tested without bearing fruit, thereby returning to the original optimizer. The best result of the final model achieved an overall accuracy score of .70, and F1 score of .63, which is the harmonic mean of the precision (positive predictive value) and sensitivity (true positive rate). In this case, it is important to have a low number of false-negative (FN) classifications. The proposed model has a false-negative classification of 45.82%, as shown in figure 3, making it too unreliable to go into production. The model is, however, capable of making the correct prediction of personal test image as shown in figure 4.

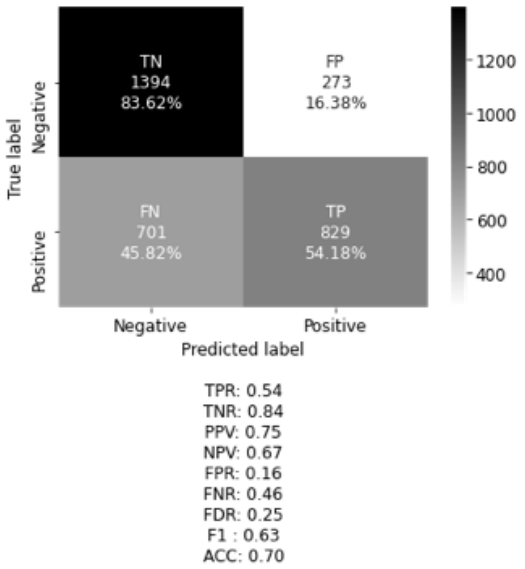


Figure 3: Confusion Matrix

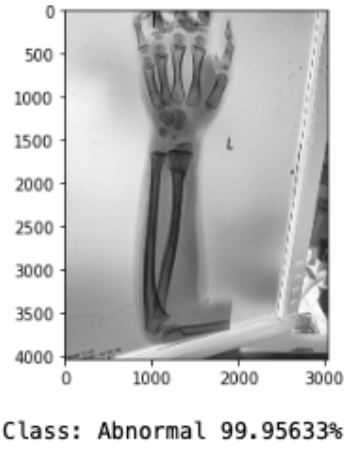


Figure 4: Personal Testing Image

6 Conclusion

A convolutional neural network has been designed, implemented, and trained on musculoskeletal radiograph images from the MURA dataset to classify bone abnormalities. The proposed model achieved an overall accuracy score of .70, and F1 score of .63. The model did not achieve a high enough accuracy to go into production; further research is needed.

References

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36:193–202, 02 1980. doi:https://doi.org/10.1007/BF00344251.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL `https://arxiv.org/pdf/1412.6980.pdf`.

Yann Lecun, Leon Bottou, Y. Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324, 12 1998. doi:10.1109/5.726791.

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, and Andrew Y. Ng. Mura: Large dataset for abnormality detection in musculoskeletal radiographs, 2018. URL `https://arxiv.org/pdf/1712.06957.pdf`.