Deep Learning in the diagnosis and therapeutic monitoring of Covid-19 patients.

FEUP & CHUSJ

August 2, 2020

1 Introduction

We present a support system for clinical diagnosis, using the Deep Learning (DL) framework to create computational models, based on convolutional neural networks architectures (ResNet , Xception , etc.), for classification of chest radiographic images. Optimize the severity assessment process determined by chest X-ray. In this context, it is possible to highlight the affected zones, through the deep autoencoders and/or the using grap-CAM algorithm on the classifier, which are the standard for anomaly detection. We use an online open data set of X-ray images. The developed APP for the user interface is also described.

In the therapeutic field we are developing a process of temporal analysis of biometric data collected from patients to formulate a model to estimate the evolution of the disease in a particular patient. This analysis will be done with models based on recurrent neural networks oriented to the study of multivariable time series: RNN, LSTM with specialized attention mechanisms.

2 X-Ray Diagnosis

2.1 Models & Data

At FEUP in collaboration with S. Joo hospital, we are developing an app for chest x-ray images diagnose. This app outputs the probability of COVID 19 infection and marks the pulmonary affected areas. We based this research on the Deep Learning framework. For these two tasks, we use a classification model and an autoencoder. We use the 2020 COVID-19 Image Data Collection [1] and from Mendeley Data Chest XRay Images for Classification [2]. The following table 1 describes the number of chest X-ray images we were able to collect.

2.2 Classification

We tested several classification models. Here we show the best results that were archived with a personalized model, trained from scratch, based on the Xception neural network architecture. In figure 2.1 is possible to see the main inception block architecture were the residual connection passes through a max-pooling layer before being concatenated to the

Table 1: Base dataset examples per class.

Class	Number of images
Normal	1583
Covid	755
Other diseases	672

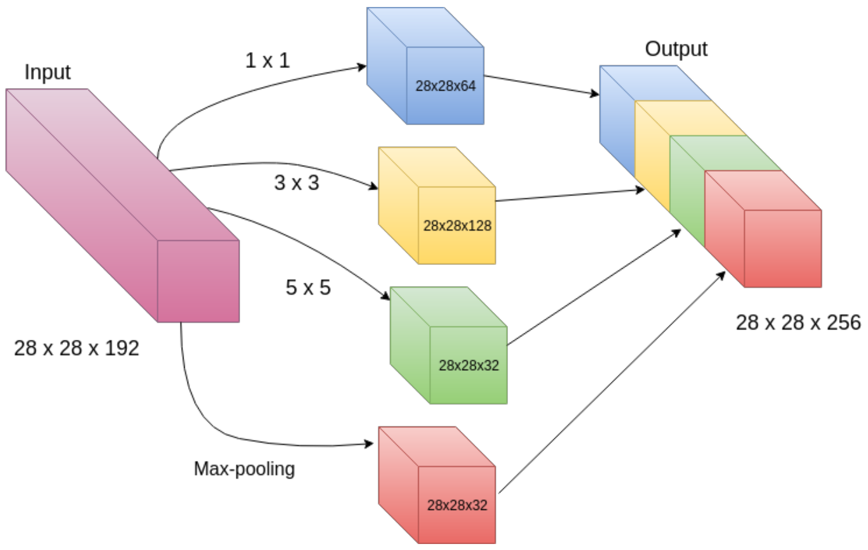

Figure 2.1: Inception main module.

output of other convolution layer processed in parallel. This module is repeated at several depths of our architecture.

2.2.1 Classifier results

The results of our classifier, on the validation set of 602 images, are presented in the following table 2 (accuracy of 97%):

Table 2: classifier.

	Precision	Recall	F1-Score	Support
Normal	0.98	0.98	0.98	317
Covid	0.98	0.99	0.98	151
Other diseases	0.94	0.94	0.94	134
AVG / Total	0.97	0.97	0.97	602

The resulting confusion matrix is presented in figure 2.2

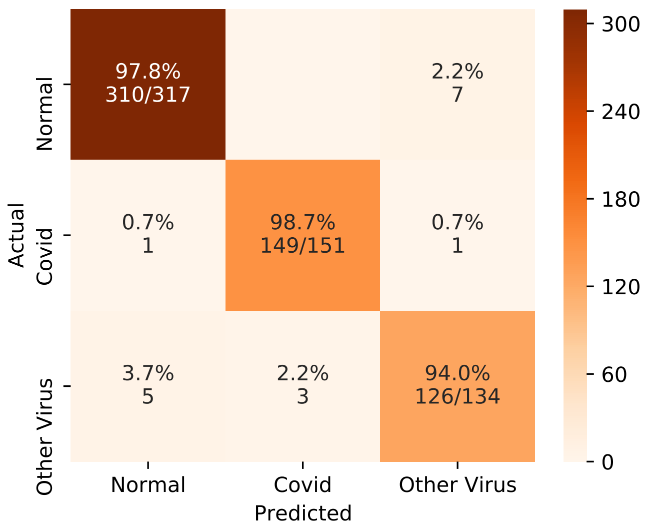

Figure 2.2: Confusion Matrix of our model.

2.3 Anomaly annotation

For the process of marking the affected areas in the image i.e. Anomaly detection/annotation, we developed and test two different approaches. The first methodology involved the creation of another DL based model for anomaly detection and marking. This model is based on a convolutional auto-encoder architecture (see figure 2.3).

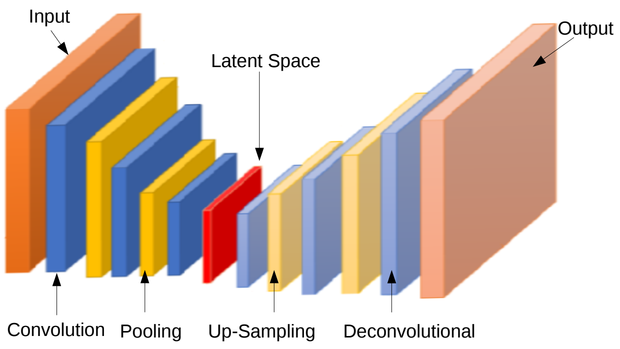

Figure 2.3: Convolutional Auto-encoder architecture.

The heatmap marking the anomalies is obtained with a convolutional auto-encoder trained with only healthy X-ray images. This model was trained with only normal class images as input and using exactly the same images as target output, for each training example. When the auto-encoder is fed with a Covid class of image, it generates a healthy image (remove the noise) based on the input. Comparing the input with the output we build a heat map of the affected areas.

The second and chosen methodology is based on the grad-CAM algorithm [4]. This algorithm aims to create visual explanations from deep networks via gradient-based localization.

This algorithm does a follow up of the activated connections from the last convolution layer of the model until the classification layer (i.e. last layer with softmax activation) for a given input. It constructs the heatmap based on the force of activation of a certain class on the coordinates of the kernel of the last convolution. Figure 2.4 illustrates the processing of grad-CAM from the last convolution layer of the model until the output layer to extract the important features localization on the input.

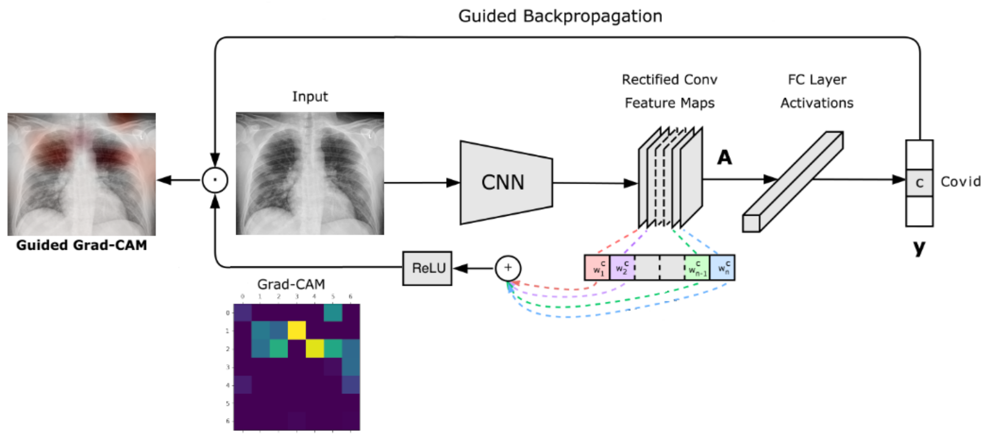

Figure 2.4: Grad-CAM algorithm.

These results are subject to validation by health professionals. We try some variations applying the grad-CAM to other convolution layers of the model instead of the last, since the last convolution layer of our model works with low resolution input maps, resulting in a low resolution heatmap. In figure 2.5 is the result of applying grad-CAM algorithm to the second before last convolution layer of our original classification model, based on Xception architecture.

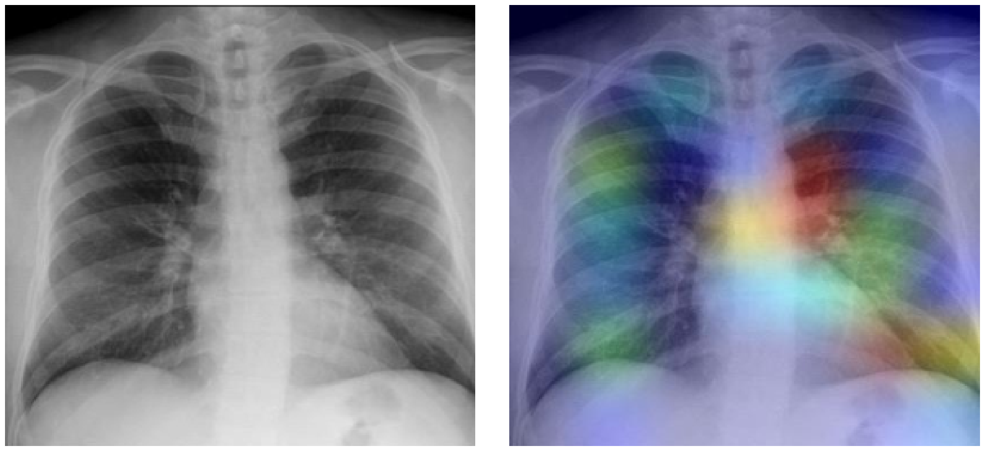

Figure 2.5: Example result of Grad-CAM algorithm using our classification model to the left image.

3 User Interface

The application that put the *models into production* was built with the Flask framework. With flask it was possible to separate both front-end and back-end improving both performance and development stability if needed in the future. The main architecture and components diagram is presented in figure 3.1.

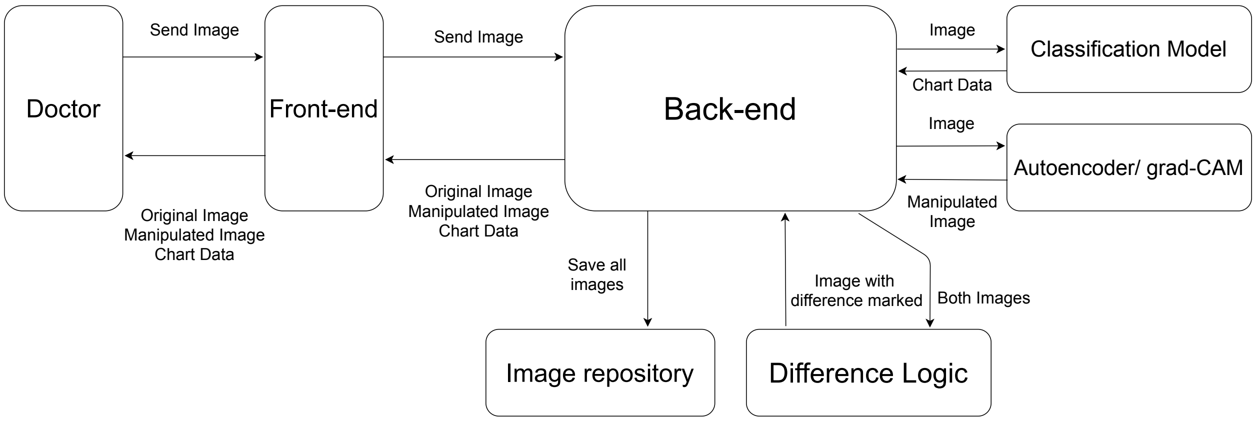

Figure 3.1: User interface aplication architecture.

This application consists of two main interfaces for the user. The first user interface is a simple upload picture form, the picture can be of any type (jpeg, png, tiff). After the upload, the image is saved in the image repository for further analysis. As we can see in the figure 3.2, the user can upload the image and wait till the loading ends at this page. then the user is redirected to the second page.

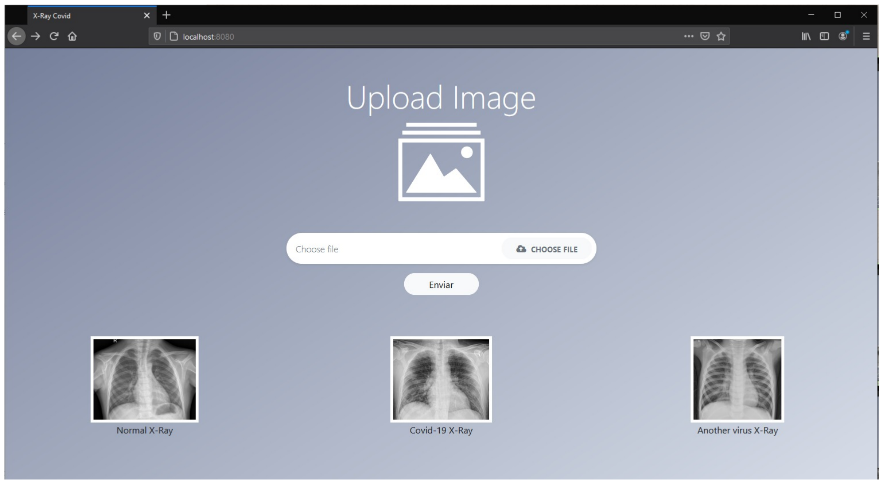

Figure 3.2: Upload Template.

In the second page the backend will first send the X-ray image to the classification model, previously trained. This model will return an array of three values representing the probability of each class. These values will then be multiplied by 100 since the values are between 0 and 1. Then the backend will populate the bar chart with the obtained values.

The bar chart has three bars one green that represents the normal x-ray, red that represents covid-19 and blue that represents other viruses (see figure 3.3). The sum of these values is 100% (i.e. result of softmax activation models last layer).

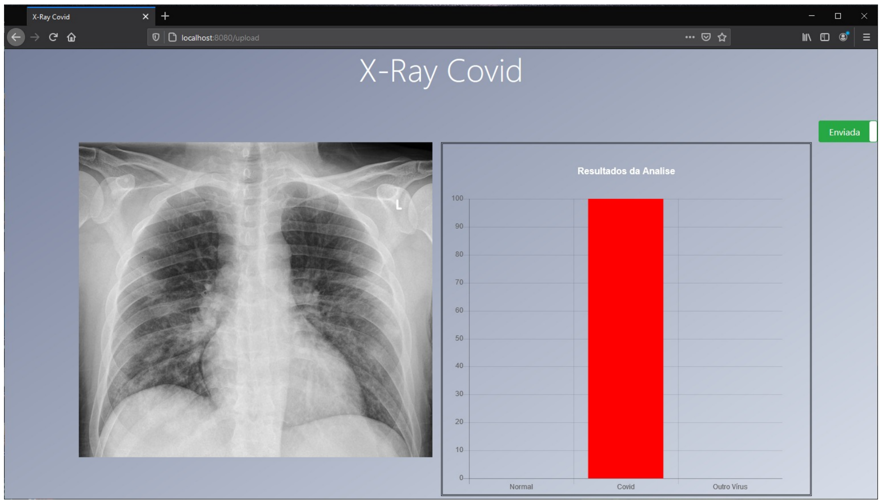

Figure 3.3: Output with covid case image.

On the left side of the page we have the uploaded picture, in this picture the user can perform zoom and pan operation by putting the mouse cursor over the image. We can also use the toggle button on the top right side to change the picture from the normal image to the annotated one. This manipulated image, with anomaly annotation, is formed in the back-end of the application by sending the original image to the autoencoder. The autoencoder changes the image to remove any signs of irregularity in the picture and then a color map is applied to the differences. Different color maps can be chosen. When the application is set to work with the grad-CAM the algorithm return directly the color map bitmap to be overlayed on the original image (see figure 3.4).

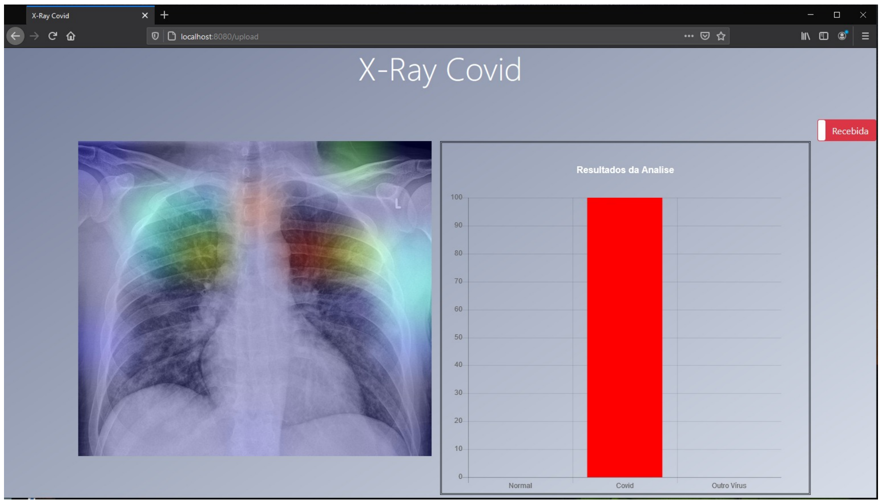

Figure 3.4: Upload Template.

References

[1] Cohen J P, MorrisonON P, DAO L, COVID-19 Image Data Collection, arXiv:2003.11597, Available online at: https://github.com/ieee8023/covid-chestxray-dataset, 2020.)

[2] D. Kermany, Zhang, and Kang, Labeled Optical Coherence Tomography (OCT), Mendeley Data, [Online]. Available: https://data.mendeley.com/datasets/rscbjbr9sj/2 , vol. 2, 2018.

[3] *François Chollet*,*Xception: Deep Learning with Depthwise Separable Convolutions*,CoRR, 2016.

[4] Grad-CAM++: Generalized Gradient-based Visual Explanations for Deep Convolutional Networks, Chattopadhyay et al, WACV, 2018