Deep Learning based on Convolution Neural Networks for the Prediction of Cerebral Malaria

[1] James Greg Mpoyi Ntambwe, [2] Eugene Mbuyi Mukendi, [3] Pierre Kafunda Katalay, [4] Simon Ntumba Badibanga.

1 Graduate in Sciences, Group: Computer Science, Option: Computer Engineering, is Head of works and doctoral student in Computer Science at the University of Kinshasa and is also Expert IT Consultant and Data base Manager in the PROSEB / World Bank projects, DR Congo.

[2] Ordinary Professor of Universities and Master of the Computer Science Chair at the Department of Mathematics and Computer Science of the University of Kinshasa, DR Congo.

[3] University Professor, initiator and head of the Optimisation and Machine Learning laboratory at the Department of Mathematics and Computer Science of the University of Kinshasa, DR Congo.

[4] University Professor, Director General of the International Bureau of Study and Research for Development, DR Congo.

Abstract:- **Given that in 2020, 96% of the 627,000 people who died of the disease were identified in Africa, according to the new analysis method of the World Health Organization (WHO). It would be necessary to reorient biomedical research in order to propose effective alternatives through predictive medicine, it is in this perspective that the present work is inscribed. Its objective is to set up a simple predictive model based on an algorithm of Convolitional Neuronal Network, implemented in the python3 environment and deployed via a Web Service, capable of analyzing and predicting the evolution or not of a malaria sufferer towards a case of severe cerebral malaria from images taken by a medical scanner. Thus, to help specialists in its treatment, to identify the pathological states of cerebral malaria at the earliest stages in order to organize the prevention by minimizing the time, the cost of the treatment and especially to avoid certain human errors related to it.**

Keywords: Deep learning, Convolutional neural network, cerebral malaria, Prediction.

1. INTRODUCTION

Coming from machine learning, deep learning has surpassed in the early 2011s all traditional methods in the processing of unstructured data, especially images, texts and sounds. Thus, there is nowadays a panorama of uses in different fields in general and in the health field in particular.

In this field, a particular type of network called "Convolutional Neural Network" was developed by Yan le Cun around the 1990s, allowing significant advances in the field of deep learning, these networks are a particular form of multilayer neural network whose architecture of connections is inspired by that of the visual cortex of mammals. Convolutional neural networks have wide applications in image and video recognition, recommendation systems and natural language processing...

To set up studies including Artificial Intelligence in order to bring more innovation for patients and improve human medical practices to save more lives. It is worth noting that the health sector is today one of the most dynamic sectors in artificial intelligence, whether in preventive medicine or medical diagnosis. This dynamic can be explained by access to large volumes of data. Thus, many hospitals are collaborating with laboratories and researchers.

The advances in Artificial Intelligence are destined to disrupt the world of health care and deep learning is now the flagship method of artificial intelligence (AI).

2. STATEMENT OF THE PROBLEM

Considering that any research project always supposes the existence of a problem to which one tries to find solutions. In this case, malaria remains the first and most deadly parasitic endemic in the world, as reported by the World Health Organization (WHO). Severe malaria can therefore take different clinical forms, the most important of which is cerebral damage (neuromalaria), untreated neuromalaria is fatal in two or three days.

The overall prognosis is essentially based on the speed of diagnosis, so to try to simplify and improve the major fact in the delay of biological diagnosis of cerebral malaria and thus avoid the late management and these potentially fatal complications. It would be better to predict on a malaria patient that his condition (the presence of vascular and hemorrhagic lesions, causing degenerative alterations of nerve cells) can evolve in a few hours to a serious malaria (neuro-malaria) and as a corollary, to reduce the socio-economic incidence of the treatment of malaria patients in the intensive care unit, because prevention is cure, as they say.

The problem thus defined, raises in us the following questioning, which we will try to solve in this work:

In order to anticipate complications related to brain damage and to reduce the mortality rate due to this parasitic endemic, what initiative should be taken in terms of computer contribution to support the medical profession in a significant way to the rapidity of the diagnosis in order to reveal already on a malaria patient a neurological component evoking the evolution towards cerebral malaria?

Starting from the computer tool, how to materialize this initiative, namely: to set up a simple and effective predictive model allowing imperatively to look for these neurological components and criteria of gravity in order to predict the evolution or not of a malaria patient towards a case of cerebral malaria?

While contributing to the thesis that Artificial Intelligence would play a crucial role in the health field, the above questions lead us to propose the use of deep learning as a means to facilitate and simplify this realization. Thus, we are confident to implement a predictive model based on convolution neural networks, which will allow learning on imaging data and a shortened prediction of the presence of the criteria of severity of brain damage by minimizing human errors. This demonstrates our innovative spirit and our ability to contribute to the solution of real problems in this crucial field in our society.

3. DEEP LEARNING AND NEURAL NETWORKS [9] [10] [11 [12]

3.1. Generalities [9] [10]

Deep learning is based on an artificial neural network, which is composed of tens or even hundreds of "layers" of neurons, each of which receives and interprets information from the previous layer. The system will learn, for example, to recognize letters before tackling the words in a text, or determine whether there is a face in a photo before discovering which person it is. You can build a network with as many layers and neurons as you want. The more layers there are, the deeper the network and the richer the model becomes, but also the more difficult it is to train.

Schematically, three main families of deep learning networks are developed with industrial ambitions by taking advantage of graphics cards (GPU) to massively parallelize the calculations at the time of learning:

- Convolutional neural networks (ConvNet) for image analysis.
- Long-short term memory (LSTM) when a temporal dimension or more generally autocorrelation properties are to be taken into account for signal processing or natural language analysis.
- Auto Encoder decoder or diabolo network in unsupervised learning for, for example, image or signal de-noising, anomaly detection.

3.2. Mathematical Notion of Convolution

Convolution is a mathematical operation widely used in Signal Processing, Probability (sum of 2 random variables); Modeling of systems via impulse response (GMC, GEL, Convolution Neural Networks).

Convolution in two dimensions is an operation that from an input matrix noted A, and a matrix of a pattern noted M, associates an output matrix A* M.

3.3. Convolutional Neural Networks [11] [12] [13].

3.3.1. Preamble

Multi Layer Perceptrons (MLP) have shown their efficiency as a learning technique for data classification. Thus, convolutional neural networks are an extension of MLPs that can effectively address the main shortcomings of MLPs. They are designed to automatically extract features from input images, are invariant to slight image distortions, and implement the notion of weight sharing allowing to considerably reduce the number of network parameters. This weight sharing also allows to take into account the local correlations contained in an image in a strong way.

A CNN generally applies 3 different types of operations to an image in order to extract the relevant information: Convolution, Pooling and ReLU-like activation function.

The first use of convolutional neural networks was made by Fukushima with his Neocognitron. The weights are forced to be equal to detect lines, points or corners at all possible locations in the image, implementing in fact the idea of weight pooling.

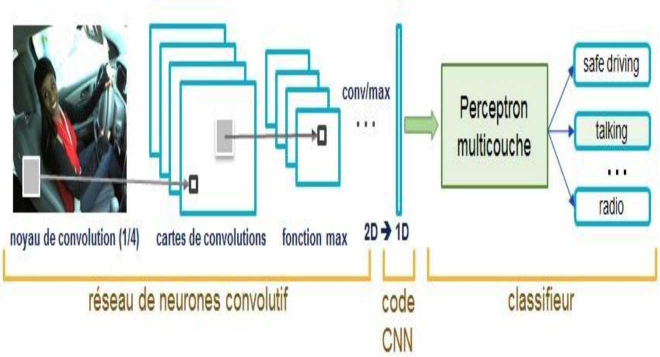


Figure III.1.Standard architecture of a neural network

3.3.1. Architecture of Convolutional Neural Networks and mode of operation

A Convolutional Neural Network is a type of acyclic (Feed-Forward) artificial NR in which the connection pattern between neurons is inspired by the visual cortex of animals. Neurons in this brain region are arranged so that they correspond to regions that overlap when the visual field passes.

An architecture of a convolutional neural network consists of two types of artificial neurons, arranged in independent layers successively processing information:

1.Processing neurons that process a limited portion of the image (called "receptive field") through a convolution function.

2.The neurons of pooling of the outputs called pooling (total or partial)

A non-linear and punctual corrective treatment can be applied between each layer to improve the relevance of the result. The set of outputs of a processing layer allows to reconstitute an intermediate image, which will be used as a basis for the following layer

1) The convolution layer (CONV)

Convolution is a fundamental operation for convolutional neural networks, it is the fundamental operation of the first layers of the network that allow to extract specific information characterizing the input A ∈ Rn×k. It is a mathematical operation that consists in applying a succession of filters W (also called convolution matrices) on a set of convolution regions of the input A ∈ Rn×k by a sliding window principle in order to produce as output a set of feature maps (also called activation maps or feature map).

Three hyperparameters are used to size the volume of the convolution layer (also called output volume):

Layer depth: number of convolution kernels (or number of neurons associated with the same receptive field);

Pitch: controls the overlap of the receptive fields. The smaller the pitch, the more the receptive fields overlap and the larger the output volume;

The margin (at 0 or zero padding): The size of this zero-padding is the third hyper parameter. This margin allows to control the spatial dimension of the output volume.

The convolution filters correspond to weights randomly initialized and then updated by the gradient back-propagation algorithm at the time of learning. The size of the filters and their directions of motion vary depending on the task at hand and the size of the convolution operation. There are 3 types of convolution operations with different dimensions 1D, 2D and 3D. The difference between these types of convolution is mainly expressed in the sizes of the filters, the number of directions of movement of the filters on the input A as well as the dimension of the feature map produced in output.

1D convolution: this convolution operation is often used in automatic language processing (in text and speech processing). The filter is characterized by a height h (corresponding to h words or h time series) and a width l which corresponds to the input width l = k. Thus, the filter can only move in one direction (from top to bottom),

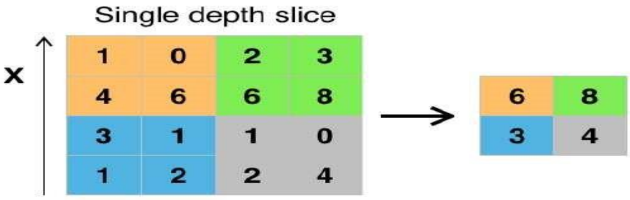
2D convolution: this operation is generally applied in image processing, where the size of a filter depends essentially on the dimension of the input. For example, for a 2D image, the filter is characterized by a height h < n and a width l < k. For a 3D image, the filter has in addition a depth f having the same size as the input f = p.

3D convolution: this operation is often used in video processing.

2) Pooling layer (POOL)

Pooling is a very powerful tool used by CNNs, ("pooling"), which is a form of image subsampling. The input image is cut into a series of non-overlapping rectangles of n pixels on each side (pooling). Each rectangle can be seen as a tile. The tile output signal is defined according to the values taken by the different pixels of the tile. Pooling reduces the spatial size of an intermediate image, thus reducing the amount of parameters and computation in the. It is therefore common to periodically insert a pooling layer between two successive convolutional layers of a CNN architecture to control overfitting (overlearning).

The most common form is a pooling layer with tiles of size 2x2 (width/height) and as output value the maximum input value. This is called "MaxPool 2x2" (compression by a factor of 4). It is possible to use other pooling functions than the maximum. You can use "average pooling" (the output is the average of the input patch values), "L2-norm pooling". In fact, even if initially the average pooling was often used, it turned out that the max-pooling was more efficient because it increases more significantly the importance of strong activations. In other circumstances, stochastic pooling can be used.



1	0	2	3
4	6	6	8
3	1	1	0
1	2	2	4

6	8
3	4

Fig. III.2. Pooling with a 2x2 filter and a step of 2

After pooling, the image has only a quarter of the number of its original pixels. Because it keeps the maximum value contained in the window at each step, it preserves the best features of that window. This means

In the end, a pooling layer is simply a pooling process on an image or a collection of images. The output will have the same number of images but each image will have a lower number of pixels. This will reduce the computational load. For example, by transforming an 8 megapixel image into a 2 megapixel image, which will make life much easier for the rest of the operations to be carried out afterwards.

3) Correction layer (ReLU):

Often, it is possible to improve the efficiency of processing by interposing between the processing layers a layer that will operate a mathematical function (activation function) on the output signals. We have in particular:

- The ReLU correction (short for Rectified Linear Units): f(x)=max (0,x). This function, also called "non-saturating activation function", increases the non-linear properties of the decision function and the whole network without affecting the receiving fields of the convolution layer;
- The hyperbolic tangent correction f(x)=tanh(x);
- The correction by the saturating hyperbolic tangent : f(x)=|tanh(x)| ;
- The correction by the sigmoid function.

Often, the Relu correction is preferable, because it results in the training of neural network several times faster, without making a significant difference to the accuracy generalization.

4) Fully connected layer FC

After several layers of convolution and max-pooling, the high-level reasoning in the neural network is done via fully connected layers. Neurons in a fully connected layer have connections to all outputs of the previous layer (as is regularly seen in regular neural networks). Their activation functions can therefore be calculated with a matrix multiplication followed by a polarization shift.

5) Loss layer (LOSS)

The loss layer specifies how the network training penalizes the deviation between the predicted and actual signal. It is normally the last layer in the network. Various loss functions adapted to different tasks can be used here. Soft max loss is used to predict a single class among K mutually exclusive classes. The sigmoid cross-entropy loss is used to predict K independent probability values in (0,1). The Euclidean loss is used to regress to real values.

3.3.2. Convolutional Processing and Image Recognition [11] [12] [13]

a) Depth 1 processing is non-overlapping (facilitating understanding)

In the context of image recognition, the latter is paved, i.e. cut into small areas called "Tiles". Each tile is processed individually by an artificial neuron (which performs a

classical filtering operation by associating a weight to each pixel of the tile).
The fact of having the same processing (same parameters), slightly shifted for each receiver field, is called "a Convolution". This stratum of neurons with the same parameters is called "convolution kernel".
The pixels of a tile are analyzed globally. In the case of an n color image a pixel contains 3 inputs (red, green and blue) which will be processed globally by each neuron. So the image can be considered as a volume, and noted for example 30*10*3 for 30 pixels of width, 10 of height and 3 of depth corresponding to the 3 red, green and blue channels. In a general way, one will speak about "volume of entry".

b) Generalization (depth and overlap)
In reality, the area analyzed is slightly larger than the tile and is called "receptive fields". The receptive fields overlap in order to obtain a better representation of the original image and a better consistency of the processing. The overlap is defined by the offset step between 2 adjacent receptive fields.
A convolution kernel will analyze a feature of the input image. To analyze several features, we will stack layers of independent convolution kernels, each layer analyzing a feature of the image.
The set of strata thus stacked forms the "convolutional processing layer" which should be seen as a volume often called "output volume". The number of processing layers is called the depth of the convolution layer (not to be confused with the depth of a convolutional neural network which counts the number of convolution layers)

3.3.4. Advantages of Convolutional Neural Networks

A major advantage of convolutional networks is the use of a unique weight associated with the signals entering all the neurons of a single convolution kernel. This method reduces the memory footprint, improves performance and allows for translation invariance. This is the main advantage of the CNN over the MLP, which considers each neuron independent and therefore assigns a different weight to each incoming signal.

When the input volume varies in time (video or sound), it becomes interesting to add a delay parameter in the neurons parameterization. In this case, we speak of a time delay neural network (TDNN).
Compared to other image classification algorithms, convolutional neural networks use relatively little preprocessing. This means that the network is responsible for evolving its own filters on its own (unsupervised learning), which is not the case for other more traditional algorithms. The absence of initial parameterization and human intervention is a major advantage of CNNs.

3.4. Some Deep Learning Algorithms

There are different Deep Learning algorithms. We can thus quote:

- Deep Neural Networks: These networks are similar to MLP networks but with more hidden layers. The increase in the number of layers allows a neural network to detect slight variations in the learning model, promoting overlearning or overfitting.
- C
onvolutional Neural Networks (CNN). The problem is divided into sub-parts, and for each part, a "cluster" of neurons will be created in order to study this specific portion. For example, for a color image, it is possible to divide the image on the width, height and depth (colors).

- The Deep Belief Network: These algorithms work according to a first unsupervised phase, followed by the classical supervised training. This unsupervised learning stage also facilitates supervised learning.

- R-CNN (Regional Neuronal Network): from an image, the modeler will extract from the image the regions most likely to contain an object. These are called areas of interest. For each of these areas of interest, a set of bourding boxes will be generated. These boxes are classified and selected according to their probability to contain the object. 2000 proposed regions are thus extracted. These regions are then received as input by the CNN. The advantages are therefore to process the image by piece and not the whole image as for a simple CNN and to be able to locate several objects in an image. It is a faster processing and less expensive in machine power.
- MaskR-CNN (Mask Regional Neuronal Network): like the previous model, allows to detect objects and to classify them. Its particularity is to add to this detection task the instance segmentation, i.e. each pixel of the image will be classified. Moreover, unlike semantic segmentation, which associates a label to each pixel, instance segmentation associates a mask and a label to each object, even if these objects belong to the same class.

It is fast, relatively simple to implement and flexible in the tasks that can be assigned to it. It has also been used in many challenges and projects related to object detection from scanner and aerial images (taken by drone or satellite).

4. CEREBRAL MALARIA [17] [18]

4.1. Concept of cerebral malaria and vector

In spite of the attention, no less legitimate, paid to the Covid-19 pandemic, the virus, transmitted by mosquito bite, still ravages poor populations. The African continent continues to pay the highest global price. In 2020, 96% of the 627,000 people who died from the disease were in Africa. WHO findings for the year 2020 reveal that it caused 69,000 additional deaths and 14 million additional infections compared to the previous year.
Cerebral malaria (neuromalaria) is the most serious, dreadful and frequent complication caused by Plasmodium falciparum infection. It consists of capillary thrombosis responsible for vascular and hemorrhagic lesions, causing degenerative alterations of nerve cells, surrounded by cellular infiltrates. There are several theories to explain these phenomena.
Malaria represents an enormous financial burden for the populations and consequently the disease constitutes an obstacle to the development of the countries concerned, especially in Africa. For all these reasons, the fight against malaria constitutes, with the fight against AIDS and

tuberculosis, one of the "Millennium Goals" defined by the United Nations, and the "Global Fund" is intended to supply the countries in need with drugs.

Another significant and historic advance is the development of the very first malaria vaccine approved by the WHO, with Gavi, the Vaccine Alliance, having given its approval for its administration to children in sub-Saharan Africa.

4.2. Diagnosis and Prognosis

The diagnosis of cerebral malaria should be considered in all patients with decreased consciousness, a history of fever, and who have been in an endemic area for 2 to 3 months prior to the onset of symptoms.

In an attempt to simplify and improve the biological diagnosis of malaria, other techniques have been developed such as rapid tests by immunochromatography on strips, rapid diagnostic tests, etc.

Cerebral malaria has a high mortality rate, between 15 and 50%. In the initial approach to the patient, there are independent factors associated with mortality

Advanced age, Presence of coma and High parasitism. The prognosis of cerebral malaria depends to a large extent on the management of complications such as renal failure, severe jaundice and metabolic acidosis.

In adults, the spectrum of presentation includes: Psychosis (4%); Cerebellar ataxia (3%); Extrapyramidal rigidity or hemiplegia; Depth of coma duration and Multiple seizures have been factors in their development.

Cognitive, language and behavioral difficulties were documented in 24% of children after cerebral malaria.

5. IMPLEMENTATION AND DEPLOYEMENT OF THE MODEL [15] [16] [20] [21] [22] [27]

It is in this last part, that we are going to invest ourselves to program our predictive model of deep neural networks CNN (Convolutional Neural Networks) in the Python3 environment.

For this, we started by defining the architecture of our model of convolutional neural networks (CNN) that we will create for the deep learning of images and then, we will apply this model on the basis of medical images taken from malaria patients.

To implement our model, we will exploit the Tensorflow and Keras libraries, our model will be based on the Mask-RCNN (Mask Regional Convolutional Neural Networks) algorithm for its learning and performance, the model is adjusted from examples, that is to say from data for which the expected result is known. Our model will therefore be able to learn to identify features (the infected area) from brain malaria data sets (images).

In order to run our model as a war machine in a reasonable time, we used our computer whose configuration is given below. It was perfectly suitable for this purpose, and after the installation of the different software bricks necessary.

5.1. Hardware and Configuration Used for this Challenge

For the implementation and realization of calculations of our model, we had made use of a laptop DELL Inspirons 15-7590 Duo CP/GPU, better to work with:

- GPU NVIDIA GeForce GTX 1650 8 GB GDDR4
- Intel Core i7 hyperthreaded cores @ 2.6GHz
- 8GB RAM
- 512 GB SSD
- OS: Duel Boot: Linux Ubuntu 21.04, 64-bit AND Windows 10 Pro 64-bit.
- Environment : Python 3.6; Keras; Tensorflow.

5.2. The Image Base

For our analysis, our data comes from researchers at the Lister Hill National Center for Biomedical Communications (LHNCBC), part of the National Library of Medicine (NLM), who carefully collected and annotated the publicly available dataset of healthy and infected blood smear images.

The following capture is an example of one of these data or medical images of brain malaria that make up our dataset.

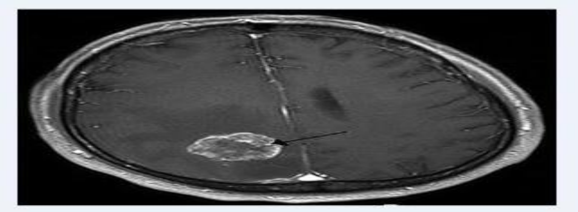


Fig. 5.1. The cerebral malaria dataset

5.3. Architecture of our Neural Network Model [20].

The architecture of the model that we present next is based on the Mask-RCNN (Mask Regional Convolutional Neural Networks) model, a Deep Learning model allowing instance segmentation: the targeted objects will be detected and classified independently from each other. Its use is objective and recommended according to our problematic, since many resources are currently available. Indeed, from images, which correspond to what the input model expects, the model will define a set of rules, allowing to achieve the prediction and thus the detection of the targeted object based on the labels and/or annotations of our dataset.

Thus, it is composed of two convolution layers, two maxpooling layers and a fully connected layer. The input image is of size 50*50, the image goes first to the first convolution layer. This layer is composed of 64 filters of size 2*2, the ReLU activation function is used, this activation function forces the neurons to return positive values, after this convolution 64 feature maps of size 50*50 will be created.

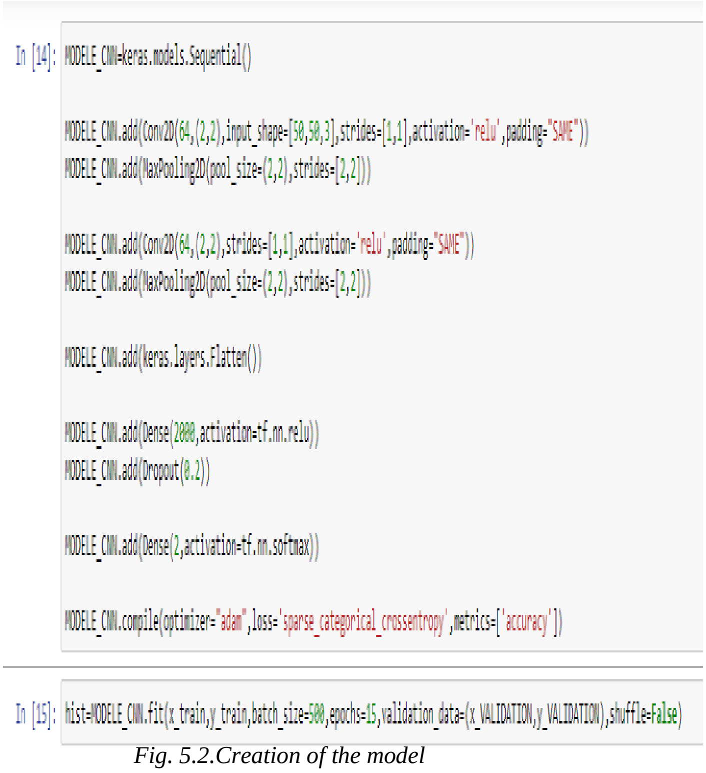
Then, the 64 feature maps that are obtained they are data in input of the second layer of convolution that is also composed of 64 filters. The ReLU activation function is applied on the convolution layers. The maxpooling is applied after to reduce the size of the image and the parameters. At the output of this layer, we will have 64 feature maps of size 25*25. The feature vector from the convolutions has a dimension of 40000.

After these two convolution layers, we use a neural network composed of a fully connected layer with a hidden layer

composed of 2500 neurons with the ReLU as activation function, and an output layer composed of two neurons with a softmax as activation function. This SOFTMAX function allows to compute the probability distribution of the 2 classes in the medical image base.
Given the depth of the network and the time needed to build it from scratch, we relied on an existing implementation in Tensorflow and Keras.

5.4. Creation and training of the model
The implementation of our model works with the Tensorflow and Keras libraries in a Python3 environment, the available implementation contains a class for the configuration of the model: the Config class. So we were going to create a class that inherits from it to configure the model for our application. The training allows the model to learn the task it is supposed to do, i.e. to detect the criteria for brain damage in a malaria patient. In our case, the model is trained in a supervised way. Once the model is trained, we must be able to evaluate its efficiency.
The following pictures show briefly the creation of our predictive model as well as the training of the model in the Python3 environment.

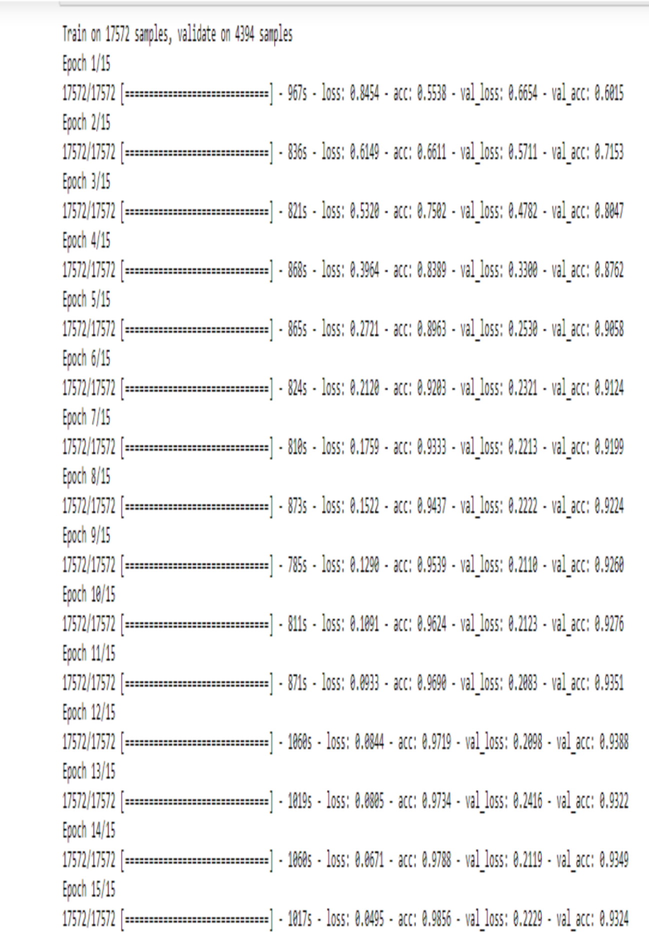
1. Creation of the model



```
In [14]: MODELE_CNN=keras.models.Sequential()


         MODELE_CNN.add(Conv2D(64,(2,2),input_shape=[50,50,3],strides=[1,1],activation='relu',padding="SAME"))
         MODELE_CNN.add(MaxPooling2D(pool_size=(2,2),strides=[2,2]))


         MODELE_CNN.add(Conv2D(64,(2,2),strides=[1,1],activation='relu',padding="SAME"))
         MODELE_CNN.add(MaxPooling2D(pool_size=(2,2),strides=[2,2]))


         MODELE_CNN.add(keras.layers.Flatten())


         MODELE_CNN.add(Dense(2000,activation=tf.nn.relu))
         MODELE_CNN.add(Dropout(0.2))


         MODELE_CNN.add(Dense(2,activation=tf.nn.softmax))


         MODELE_CNN.compile(optimizer="adam",loss='sparse_categorical_crossentropy',metrics=['accuracy'])

In [15]: hist=MODELE_CNN.fit(x_train,y_train,batch_size=500,epochs=15,validation_data=(x_VALIDATION,y_VALIDATION),shuffle=False)
```

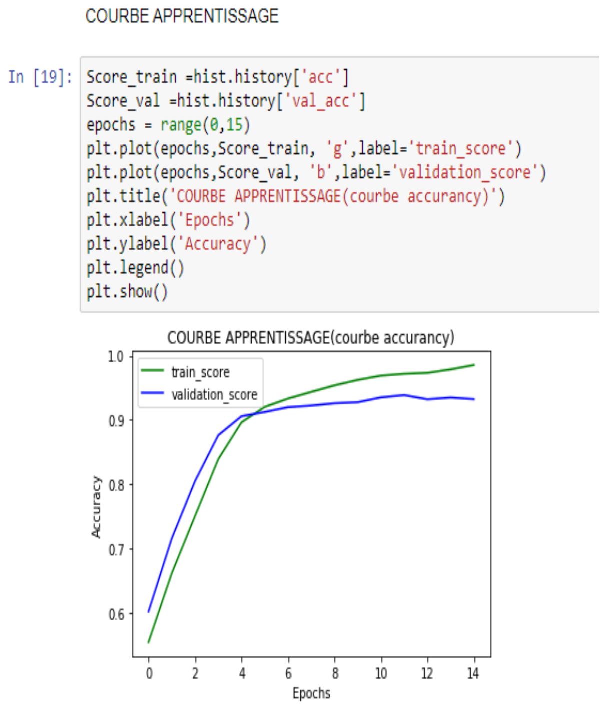
Fig. 5.2.Creation of the model

2. Training of the model



```
Train on 17572 samples, validate on 4394 samples
Epoch 1/15
17572/17572 [==============================] - 967s - loss: 0.8454 - acc: 0.5538 - val_loss: 0.6654 - val_acc: 0.6015
Epoch 2/15
17572/17572 [==============================] - 836s - loss: 0.6149 - acc: 0.6611 - val_loss: 0.5711 - val_acc: 0.7153
Epoch 3/15
17572/17572 [==============================] - 821s - loss: 0.5320 - acc: 0.7502 - val_loss: 0.4782 - val_acc: 0.8047
Epoch 4/15
17572/17572 [==============================] - 868s - loss: 0.3964 - acc: 0.8389 - val_loss: 0.3300 - val_acc: 0.8762
Epoch 5/15
17572/17572 [==============================] - 865s - loss: 0.2721 - acc: 0.8963 - val_loss: 0.2530 - val_acc: 0.9058
Epoch 6/15
17572/17572 [==============================] - 824s - loss: 0.2120 - acc: 0.9203 - val_loss: 0.2321 - val_acc: 0.9124
Epoch 7/15
17572/17572 [==============================] - 810s - loss: 0.1759 - acc: 0.9333 - val_loss: 0.2213 - val_acc: 0.9199
Epoch 8/15
17572/17572 [==============================] - 873s - loss: 0.1522 - acc: 0.9437 - val_loss: 0.2222 - val_acc: 0.9224
Epoch 9/15
17572/17572 [==============================] - 785s - loss: 0.1290 - acc: 0.9539 - val_loss: 0.2110 - val_acc: 0.9260
Epoch 10/15
17572/17572 [==============================] - 811s - loss: 0.1091 - acc: 0.9624 - val_loss: 0.2123 - val_acc: 0.9276
Epoch 11/15
17572/17572 [==============================] - 871s - loss: 0.0933 - acc: 0.9690 - val_loss: 0.2083 - val_acc: 0.9351
Epoch 12/15
17572/17572 [==============================] - 1060s - loss: 0.0844 - acc: 0.9719 - val_loss: 0.2098 - val_acc: 0.9388
Epoch 13/15
17572/17572 [==============================] - 1019s - loss: 0.0805 - acc: 0.9734 - val_loss: 0.2416 - val_acc: 0.9322
Epoch 14/15
17572/17572 [==============================] - 1060s - loss: 0.0671 - acc: 0.9788 - val_loss: 0.2119 - val_acc: 0.9349
Epoch 15/15
17572/17572 [==============================] - 1017s - loss: 0.0495 - acc: 0.9856 - val_loss: 0.2229 - val_acc: 0.9324
```

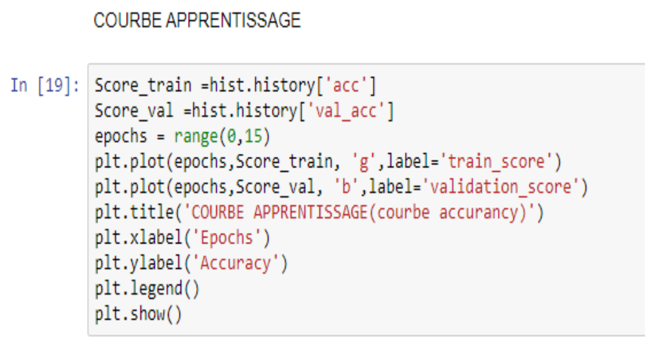
Fig. 5.3.Training of the model



COURBE APPRENTISSAGE

```
In [19]: Score_train =hist.history['acc']
         Score_val =hist.history['val_acc']
         epochs = range(0,15)
         plt.plot(epochs,Score_train, 'g',label='train_score')
         plt.plot(epochs,Score_val, 'b',label='validation_score')
         plt.title('COURBE APPRENTISSAGE(courbe accurancy)')
         plt.xlabel('Epochs')
         plt.ylabel('Accuracy')
         plt.legend()
         plt.show()
```

Fig. 5.4. The learning curve (loss)



COURBE APPRENTISSAGE

```
In [19]: Score_train =hist.history['acc']
         Score_val =hist.history['val_acc']
         epochs = range(0,15)
         plt.plot(epochs,Score_train, 'g',label='train_score')
         plt.plot(epochs,Score_val, 'b',label='validation_score')
         plt.title('COURBE APPRENTISSAGE(courbe accurancy)')
         plt.xlabel('Epochs')
         plt.ylabel('Accuracy')
         plt.legend()
         plt.show()
```

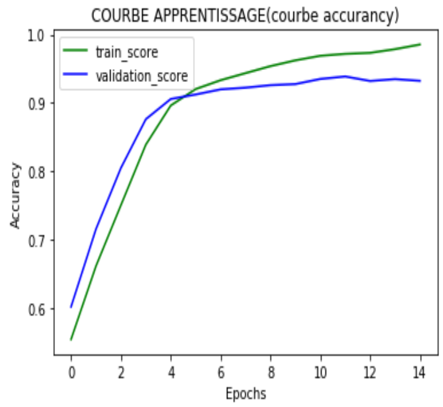


Fig. 5.5. The learning curve (accuracy)

After the analysis of the results obtained, we note the following remarks:

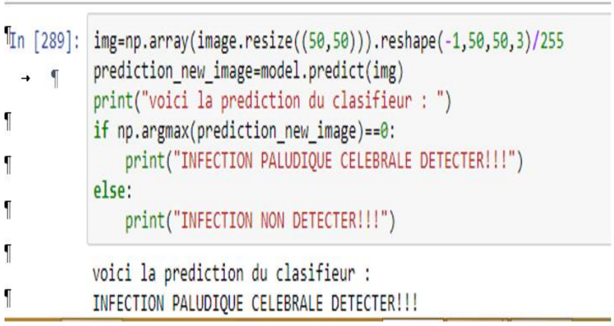
According to Figure 5.4, the precision of the learning and validation increases with the number of epochs, this reflects that at each epoch the model learns more information. If the accuracy is decreased then we will need more information to make our model learn and therefore we must increase the number of epochs and vice versa. The model has a very good performance of 93%.

From the figures above, the learning and validation error decreases with the number of epochs and the model generalizes well over the validation set.

5.4. Prediction of a New Case by the Shape Model.

With our pre-trained model, it is possible to launch predictions i.e. to provide an image to the model and to obtain in result the detection and localization of the criteria (objects) searched.

The following picture explains the prediction of a new case by our shape model.



```
In [289]: img=np.array(image.resize((50,50))).reshape(-1,50,50,3)/255
          prediction_new_image=model.predict(img)
          print("voici la prediction du clasifieur : ")
          if np.argmax(prediction_new_image)==0:
              print("INFECTION PALUDIQUE CELEBRALE DETECTER!!!")
          else:
              print("INFECTION NON DETECTER!!!")

          voici la prediction du clasifieur :
          INFECTION PALUDIQUE CELEBRALE DETECTER!!!
```

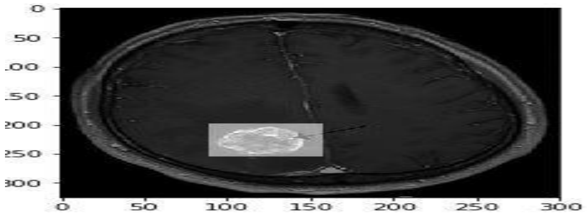


Fig. 5.6. Prediction of a new case

5.5. Deployment of our Model

For the optimal deployment of our predictive model, we opted for the implementation of a Web Service, which is a standardized medium allowing the communication and the exchange of data between heterogeneous applications and systems in a distributed environment. For this purpose, to make it simple, we have installed a SOAPUI.5.6.1 environment which is an Open source application allowing the testing of web services in a service oriented architecture. The creation of our model deployment application whose front end or presentation layer is developed in the NetBeans IDE8.1 environment.

We have deployed this User Interface to be as simple and user friendly as possible with a single button for prediction and an IMAGE/IRM ID window to enter the image/IRM number of a patient as stored in our application server.

With this in mind, the following screenshots clearly show the User Interface of our predictive model as it is deployed before and after prediction.

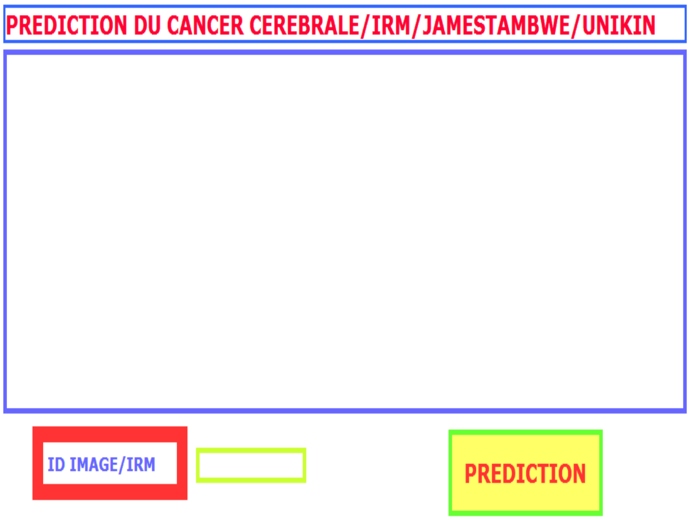


Fig. 5.7. User Interface before prediction

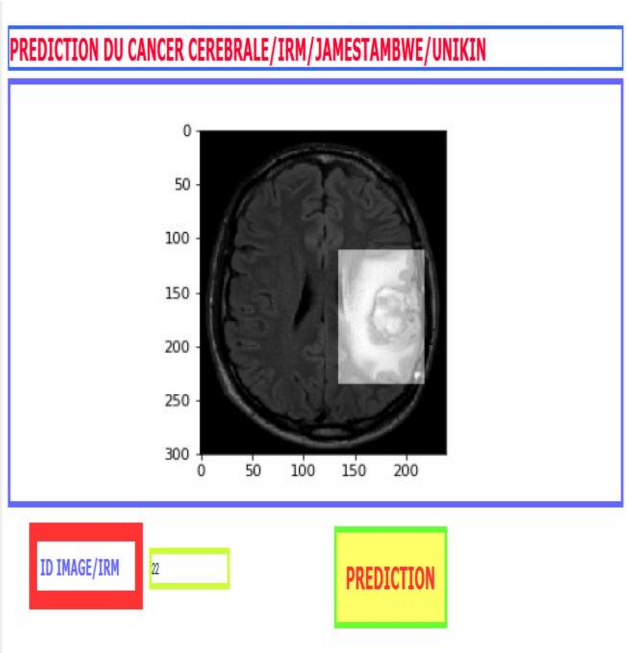


Fig. 5.8. User Interface after

6. CONCLUSION

During this challenge, which focused on the design and deployment of a predictive model in deep learning based on convolutional neural networks in order to predict after analysis and detection on an image from a scanner of a malaria patient that his condition (the presence of vascular and hemorrhagic lesions, causing degenerative alterations of nerve cells) can evolve in a few hours to a severe malaria (neuro malaria). Thus, helping treating physicians to better manage with time saving by avoiding human errors that often occur during this process as well as the late management of potentially very deadly cerebral malaria cases.

Thus, an input data, in this case an image captured by the medical scanner on a malaria patient will go through the different layers of the neural network of our model that learns to extract features, until it allows to output the result, that is to say to predict if the person will suffer from cerebral malaria or not. We used a real learning dataset made up of elements collected by researchers from the Lister Hill National Center for Biomedical Communications (LHNCBC), which is part of the National Library of Medicine. This, thanks to the depth of convolutional neural networks that process complex data, and very high dimensional with high predictive accuracy, we justify the choice. And also its ability to process unstructured data (images, videos, etc.) with more accuracy than the classical machine learning approach.

Our strategic approach led us to design and develop our model using Python (Keras, Tensorflow and M-RCNN) and we showed the results obtained in terms of accuracy and error. The results found show that the number of epochs, the size of the base and the depth of the network are important factors for obtaining better results.

In the implementation and deployment phase, we encountered some problems, the use of a CPU made the execution time too expensive at about 20,000 seconds. Nevertheless, we used deeper convolutional neural networks deployed on a CUDA GPU and NVIDIA cuDNN instead of a CPU on larger bases, to solve the said problem.

The crux of our work remains its impact in order to propose effective alternatives through predictive medicine, whose objectives will be to identify pathological conditions at the earliest stages in order to organize prevention and minimize the cost of treatment, as a corollary to reduce the very high mortality rate that it causes and which the African continues to pay the heaviest toll. Thus demonstrating our innovative spirit and our ability to propose solutions to real problems in the health field, because the doctor will have within his reach a powerful diagnostic aid, unquestionably changing his medical practices.

This work attempts to make a vital scientific contribution in this crucial area, and is therefore another stone in the various international and national projects combining clinical research and artificial intelligence. Not with the objective of replacing the doctor by the machine, but to accompany him in his heavy tasks. It is an approach that also helps to promote good diagnoses and to fight against medical errors by generating fast and accurate diagnoses.

Considering the results to which our research has led, let us be allowed to maintain that the objectives it has set itself have been perfectly achieved. Especially in the current context, where the tools for the prediction of the different pathologies are not yet numerous in our health system, and therefore, our work is of an imperious necessity.

Thanking the possible readers, we emphasize that this work is open to any improvement, as we do not pretend to have reached perfection.

7. REFERENCES

[1] Yasmine CHIADMI, Master thesis, the Impact of Artificial Intelligence in clinical research, University of Lille, Faculty of Engineering and Health Management (September 2019).

[2] BPI France Nicolas GREMY, Panorama of French health start-ups using AI, (accessed on 28/06/2020), available at https://blog.lehub.bpifrance.fr/panoramastartups-sante-francaises-ia/

[3] https://www.datacademie.com/blog/le-big-data-et-lintelligenceartificielle/

[4] Prof. Eugene Mbuyi Mukendi, Artificial Intelligence In-depth, Doctoral Seminar in Computer Science at the Faculty/Science Department/Math-info 2019-2020..

[5] Prof. Pierre Kafunda Katalay, Automatic Classification, Doctoral dissertation in faculty/science department/ Math-info academic year 2019-2020.

[6] Julien Ah-Pine, Apprentissage automatique (julien.ah-pine@univ-lyon2.fr) Université de Lyon M2 DM 2018/2019.

[7] G.Saint-Cirgue « apprendre le Machine Learning en une semaine »,*2019* (http://Machinelearnia.com)).

[8] https://msdn.microsoft.com/big_data_france/2014/06/17/evaluer-un-modle-en-apprentissageautomatique/

[9] LISA Lab, Convolutional Neural Networks (LeNet) – DeepLearning 0.1 documentation » (http://deeplearning.net/tutorial/lenet.html), (consulted on September 02, 2021).

[10] Yann LeCun, Deep Learning: Course at Collège de France in 2016 college-de-france.fr/site/en-yann-lecun/inaugural-lecture-2016-02-04-18h00.htm.

[11] Philippe LerayPhilippe, Quelques Types de Réseaux de Neurones, La Rétropropagation. Leray@insa-rouen.fr INSA Rouen -Départment ASI.

[12] I. Sutskever, A. Krizhevsky, et G. E. Hinton, « ImageNet Classification withDeep Convolutional Neural Networks », Advances in neural Processing Systems of treatment, vol. 1, 2012, p. 1097–1105 (read online (http://papers.nips.cc/paper/4824-IMAGEnet)

[13] Y. Bengio, Cours IFT6265 « Poids Partagés, TDNNs et réseaux à convolution », Université de Montréal-Département d'informatique et de recherche opérationnelle,http://www.iro.umontreal.ca/~bengioy/ift6265/convo/convo.htm juillet 2018.

[14] A. Krizhevsky, I. Sutskever et G. E. Hinton. « ImageNet Classification with Deep ConvolutionalNeural Networks », Advances in neural Processing Systems de traitement. 2012.

[15] Loussaief S, Abdelkrim A. "Machine learning framework for image classification," 2016 7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Hammamet, 2016, pp. 58-

[16] Lucie Camanez ; Extraction d’objets pour la cartographie par deep learning à partir d’images aériennes ou satellitaire, article publie le 04 /06/2020. https://www.makina-corpus.com/

[17] F. Vachon, C. Carbon, C. Gibert « Diagnostic et traitement du Paludisme Cérébral en zone endémique » (consulté le 12/04/2019), disponible sur : Bul.org.mond.santé.

[18] https://fr.wikipedia.org/wiki/R%C3%A9seau_neuronal_convolutif

[19] https://www.emconsulte.com/article/8808/traitement-du-paludisme-cerebral-de-l-enfant-afric

[20] https://www.actuia.com/contribution/cyrille-kone/a-travers-les-reseaux-de- neurones-a-convolution-en-deep-learning/

[21] https://www.actuia.com/contribution/cyrille-kone/a-travers-les-reseaux-de-neurones-a-convolution-en-deep-learning