A Detailed Study on the Choice of Hyperparameters for Transfer Learning in Covid-19 Image Datasets using Bayesian Optimization

Miguel Miranda[1], Kid Valeriano[2], José Sulla-Torres[3]
Universidad Nacional de San Agustín de Arequipa, Perú

***Abstract*—For many years, the area of health care has evolved, mainly using medical images to detect and evaluate diseases. Nowadays, the world is going through a pandemic due to COVID-19, causing a severe effect on the health system and the global economy. Researchers, both in health and in different areas, are focused on improving and providing various alternatives for rapid and more effective detection of this disease. The main objective of this study is to automatically explore as many configurations as possible to recommend a smaller starting hyperparameter space. Because the manual selection of these hyperparameters can lose configurations that generate more efficient models, for this, we present the MKCovid-19 workflow, which uses chest x-ray images of patients with COVID-19. We use knowledge transfer based on convolutional neural networks and Bayes optimization. A detailed study was conducted with different amounts of training data. This automatic selection of hyperparameters allowed us to find a robust model with an accuracy of 98% in test data.**

Keywords*—Transfer Learning; COVID-19; X-ray image; deep learning; Bayes optimization; machine learning; hyperparameter optimization

I. Introduction

Currently, there are different areas of knowledge where algorithms, heuristics, and artificial intelligence models are applied. The area of health care has evolved year after year, adopting the use of technology to save lives and improve life quality. Mainly visual information is frequently applied for the detection and evaluation of diseases. For that reason, the established fields of computer vision and medical imaging provide essential tools. The integration of these technologies and data analysis from various sources, real-time processing are core competencies necessary for the successful improvement of healthcare systems.

As we know, at the end of 2019, the first official cases of COVID-19 began to be reported in China, which continues to wreak havoc around the world today, mainly in the health system and the global economy. Currently, around 113 million cases of coronavirus (SARS-CoV-2) have been registered globally, 87 million cases of patients cured of COVID-19 and with a mortality of more than 2 million people[1]. This virus is an epidemic disease, which can cause respiratory infections from a cold to serious illnesses such as Middle East Respiratory Syndrome (MERS) and Severe Acute Respiratory Syndrome (SARS).

[1]https://www.statista.com/statistics/1087466/covid19-cases-recoveries-deaths-worldwide

In order to better control the problems caused by COVID-19 and help reduce the death rate, the detection of this disease through medical imaging is an important factor. Currently, Chest X-rays (CXR) and Computed Tomography (CT) are commonly used, which allow the severity of the disease in the patient to be assessed and monitored.

At the beginning of the pandemic, at least in Latin American countries, such as Peru, serological and molecular tests were in short supply. They were caused by the difficulty of acquiring their central governments due to the high demand for purchase by the other countries. In these countries, due to the paucity of tests, chest radiographs were used more frequently; these were used because they are more accessible to patients. However, this presents a challenge for radiologists since pneumonia can be caused by other viruses or bacteria, making it difficult to diagnose and predict Covid-19 in the patient.

Nowadays, to face this challenge, computer-aided diagnostics (CAD) is used, accelerating and improving medical diagnosis precision. As part of this problem's solution, artificial intelligence algorithms are used due to the large-scale data processing capacity integrated into CAD. For this case, an analysis of medical images and deep learning is carried out. Specifically, transfer learning is used with models based on convolutional neural networks (CNN). Some of these models already used for related studies are RESNET50[1], COVIDNET[2], ResNet50V2 [3], VGG16 [4].

At the beginning of the pandemic, training data was not available, this being the limitation even though the authors used data augmentation techniques. In this work, data are collected from various sources to make an extensive data set for further study. Besides, Bayes optimization is used to carry out more in-depth research, such as the effect of using different amounts of data, the batch size, and other adjustments. Finally, it is studied and recommended that pre-trained models better adhere to the use of medical images to detect covid. This whole study aims to help choose the most optimal values of the hyperparameters that will directly influence the performance of these models.

All the above can help improve performance and speed to detect cases of Covid-19, as well as other diseases that can affect the lungs. This can be done by finding the best model for this type of "problem". For this study, Transfer Learning is used, explained, and detailed in this article's content, where the results obtained are shown.

After this introduction, this article is organized as follows: The works related to this article are explained in Section II,

the methods and material used in this research in Section III, the methodology in Section IV, and the experimental setups in Section V, experimental result in Section VI, conclusions in Section VII,and finally, future works in Section VIII.

II. Related Work

There are works such as [5], [6], and [7], which use medical images, such as images captured from colonoscopy videos, endoscopies for the diagnosis of different gastrointestinal diseases. If we refer specifically to the diagnosis of conditions that affect the lungs. We have as a reference [8], where they used images to classify different diseases that occur in the lungs, such as pneumonia, sarcoidosis, and cancer, obtaining in each case more than 78% accuracy. Another similar work is [9], where they classify the two types of pneumonia, either of viral or bacterial origin; for this, the authors divide their methodology into three steps: first, they segment the critical part of the lung, that is, the left and right area of the lung, then they extract the characteristics of each image making use of transfer learning, and finally they use those characteristics for binary classification using the SVM algorithm.

The first works for this task used less amount of data since it was not available due to the virus's recent appearance. Among the most used techniques are convolutional neural networks (CNN). An excellent example of Convolutional Neural Networks' application is the work [10], which could obtain a precision of 85% despite the data limitation. It should be noted that this work was only based on one model, SqueezeNet, compared to the proposed work that is used up to 6 models.

Like [11] and [10], the works are both based on convolutional neural networks, but at this time, due to the little data available, data augmentation was used to have more training data. The work [11] divides work into three phases: a data augmentation phase, the second feature extract phase using already trained CNN models based on transfer learning, and finally, a classification phase.

In other works referred to as [12], carried out a few weeks after starting this pandemic, due to the lack of data, they used algorithms such as Generative Adversarial Networks (GAN) to generate images with positive Covid-19 cases. These data were used to train models, validate them, and diagnose the disease from the generated models. Despite using a GAN instead of data augmentation, the final precision obtained was 77%.

The use of some automatic hyperparameter optimization approach is essential here. The CNN needs adjustments of several hyperparameters that directly affect the model's performance. In-state of the art, they recommend using Bayes optimization, having as its main characteristic the consideration of past iterations, which is why it was chosen to follow this approach. However, there are tools and libraries such as Auto-Weka [13], Auto-Keras [14], and Google Vizier [15] that promise this automatic optimization of hyperparameters. But they do not have the flexibility to consider, for example, batch_size, pre-trained models, or some other characteristic that could be considered as hyperparameters.

Taking as reference the work [16], wherein the same way it uses this Bayes approach for the automatic optimization of hyperparameters considering the pre-entered models and the descent of gradients' optimization function. However, recently in work [17], they consider that the Batch Size is an essential hyperparameter in the classification of medical images using convolutional networks. They conclude that a large batch size does not necessarily produce better accuracy. Inspired by these studies, it was considered as optimization space: pre-trained models, the optimization function of the descent of gradients, learning rate, momentum, and batch size as hyperparameters.

III. Methods and Materials

This section describes how the data were collected and the sources from which they were obtained, and their characteristics. Besides, this work's theoretical concepts are explained in a didactic way, such as the deep learning algorithms, the transfer learning process, and the automatic hyperparameter optimization algorithm.

A. Dataset

For the present study, we collected and used a dataset of chest X-ray radiography (CXR) images acquired from various publicly available medical repositories [18] [19] [20] hosted on Kaggle. At the beginning of the pandemic, these data were scarce, wherein most of the studies used data augmentation to achieve a greater amount of data, both for training and validation.

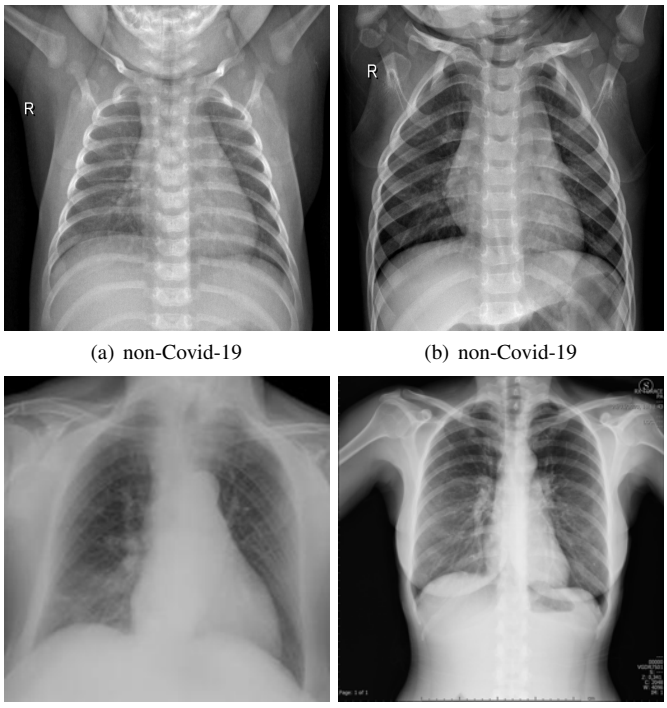
This repository was developed by a team of researchers from the University of Qatar, Doha, Qatar, Dhaka, Bangladesh, and their collaborators from Pakistan and Malaysia in collaboration with doctors. They created a chest X-ray images database for COVID-19 positive cases along with Normal and Viral Pneumonia Images. COVID data is collected from different public access data sets, online sources, and published articles. All images are in Portable Network Graphics (PNG) file format and 1024 * 1024 pixels and 256 * 256 pixels. As shown in Fig. 1, contain images for the two class (Covid-19 and Non-Covid-19).

The dataset used in this study includes a total of 5,641 2D X-ray images in the posteroanterior (PA) view of the chest. There was an unbalanced distribution of classes, 1300 images labeled as covid, and 4341 as non-covid. In order not to bias our model, 1,300 images were randomly extracted from each class, specifically in the non-covid surplus class as shown in Figure x, contain images for the two class (Covid-19 and Non-Covid-19). The test dataset used in this work was collected by [21], which is intended to simulate the real world. It is also necessary to highlight that this data is not within the training data. This test dataset contains 5000 non-covid images and 184 images categorized as covid-19.

B. Convolutional Neural Networks (CNN)

The convolutional neural network (CNN) is a deep learning neural network class. In short, think of CNN as a machine learning algorithm that can take an input image, assign importance (weights and learnable biases) to various aspects/objects in the image, and differentiate one from the other. CNN works by extracting features from images. Any CNN consists of the following:

- The input layer is a grayscale image.



(a) non-Covid-19 (b) non-Covid-19

(c) Covid-19 (d) Covid-19

Fig. 1. Example Images from Each Class Covid-19 and non-Covid-19.

- The output layer, which is a binary or multi-class label.
- Hidden layers consisting of convolution layers, ReLU (rectified linear unit) layers, grouping layers, and a fully connected neural network.

It is essential to understand that artificial neural networks, made up of multiple neurons, cannot extract features from the image. This is where a combination of convolution and grouping layers comes into play. Similarly, convolution and grouping layers cannot perform classification, so we need a fully connected neural network.

1) Convolution: In the Convolution, product and sums operations are performed between the starting layer and the n filters (or kernel) that generate a characteristic map. The extracted characteristics correspond to each possible location of the filter in the original image. The advantage is that the same filter (= neuron) is used to extract the same characteristic in any part of the input, with this that manages to reduce the number of connections and the number of parameters to train compared to a multilayer network of real connection.

After applying the Convolution, an activation function is applied to the feature maps. The recommended activation function is sigmoid ReLU, selecting a suitable learning rate and monitoring the fraction of dead neurons; it could also be tried with Leaky ReLu or Maxout, but never use logistic sigmoid.

2) Reduction: In Reduction, the number of parameters is reduced by keeping the most common characteristics. The way to reduce parameters is done by extracting statistics such as the average or maximum of a fixed region of the characteristics map; when reducing characteristics, the method loses precision although its compatibility improves.

3) Sorter: At the end of the convolutional and Reduction layers, it is often used fully connected layers in each pixel is considered as an individual neuron as in a multilayer perceptron. The last layer of this network is a classifier layer that will have as many neurons as the number of classes to predict.

C. Transfer Learning

Transfer Learning is the act of transferring knowledge from one network to another or, in general, from one model to another as shown in Fig. 2. Convolutional neural networks, as we know, stand out for learning, on their own, to interpret the images that we pass to them. In other words, they are experts in creating high-quality features. For this reason, pre-trained convolutional neural networks are, in many cases, the ideal feature extractors. We give the definitions of "domain" and "task", respectively.

A domain D consists of two components: a feature space X and a marginal probability distribution $P(X)$, where $X = x_1, ..., x_n \in X$. For example, if our learning task is to classify documents, each term is considered a binary characteristic. Hence, X is the space of all vectors of terms, x_i is the ith vector of terms corresponding to some documents, and X is a particular learning sample. In general, if two domains are different, then they may have different feature spaces or different marginal probability distributions.

Here is a unified definition of transfer learning. Definition 1 (Transfer Learning). Given a source domain D_s and a learning task T_s, a target domain D_T, and a learning task T_T, transfer learning aims to help improve the learning of the target predictive function f_T in D_t using the knowledge in D_s y T_s, where $D_S \neq D_T$, or $T_s \neq T_T$.

In the above definition, a domain is a pair $D = X, P(X)$. Therefore, the $D_S \neq D_T$ condition implies that $X_S \neq X_T$ or $P_S(X) \neq P_T(X)$. For example, in our document classification example, between a set of source documents and a set of destination documents, the term's characteristics are different between the two sets (they use different languages), or their marginal distributions are different.

Similarly, a task is defined as a pair $T = , P(Y|X)$. Therefore, the $T_S \neq T_T$ condition implies that $Y_S \neq Y_T$ or $P(Y_S|X_S) \neq P(Y_T|X_T)$. When the destination and source domains are the same, D_S = D_T, and their learning tasks are the same, that is, $T_S = T_T$, the learning problem becomes a traditional machine learning problem. When the domains are different, then 1) the feature spaces between the domains are different, that is, $X_S \neq X_T$, or 2) the feature spaces between the domains are the same, but the marginal probability distributions between the domain data are different; that is, $P(X_S) \neq P(X_T)$, where $X_{S_i} \in X_S$ and $X_{T_i} \in X_T$. For example, in our document classification example, case one corresponds to when the two sets of documents are described in different languages. Case two may correspond to when the source domain documents and destination domain documents focus on different topics. Given the specific domains D_S and D_T, when the learning tasks T_S and T_T are different, then

1) the label spaces between the domains are different, that is, $Y_S \neq Y_T$, or 2) the conditional probability distributions between the domains are different; that is, $P(Y_S|X_S) \neq P(Y_T|X_T)$, where $Y_{S_i} \in Y_S$ and $Y_{T_i} \in Y_T$. In our document classification example, case 1 corresponds to the source domain has binary document classes, while the destination domain has ten classes to classify the documents. Case 2 corresponds to the situation where the source and destination documents are very unbalanced in terms of the user-defined classes.

Furthermore, when there is some relationship, explicit or implicit, between the two domains' feature spaces, we say that the source and destination domains are related.

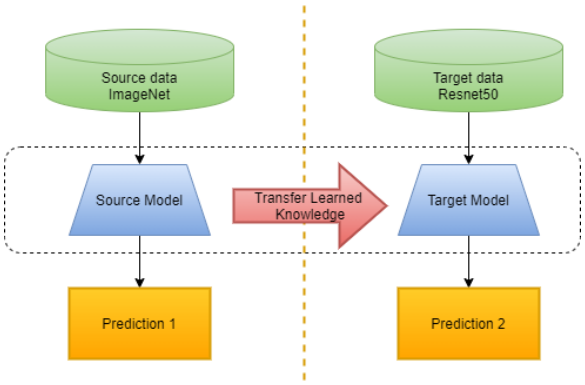


Fig. 2. Transfer Learning Workflow.

D. Bayes Optimization Algorithm

Bayesian optimization is a technique used to optimize an objective function $f(x)$, also called latent or underlying [22]. Its application is made in scenarios where the observations have a high cost, can have noise, and there is no expression for $f(x)$. Given these three characteristics, the objective is to obtain the values that, in addition to providing the most significant amount of information, minimize the objective function in the least possible number of observations.

Scenarios can be proposed in artificial intelligence, such as synthesizing a molecule with certain characteristics, where each evaluation may require a real costly experiment in time and money. In this case, we want to obtain the best parameter configuration, given by a vector x, that obtains the smallest error in $f(x)$. Therefore the mathematical expression that collects this would be the following

$$x_* = \min f(x) \tag{1}$$

Where x is the input value, which minimizes the underlying function $f(x)$, and X is the feature space where it is being optimized. If X is chosen wrong, $f(x)$ is not optimized correctly. In order to effect this minimization, a trade-off will be made between exploiting promising solutions and exploring unknown areas of the entry space. To achieve this trade-off, the acquisition function will use the mean $\mu(x_n + 1)$ and the covariance $\sigma^2(x_n + 1)$. To thus calculate the expected utility of observing a certain point x_n+1. Therefore, Bayesian optimization is a technique that makes its predictions based on the belief that one has about the model. This approach to problems achieves better results than a simple random search or grid search since neither of these strategies uses the model to guide the minimization process.

The Bayesian optimization pseudocode follows:

1) Set $t \leftarrow 0$ randomly generate the initial population $P(0)$
2) Select a promising string set $S(t)$ from $P(t)$
3) Build network B using a chosen metric and constraints
4) Generate a set of new chains $O(t)$ according to the joint distribution encoded by B
5) Create a new population $P(t+1)$ by replacing some strings of $P(t)$ with $O(t)$ set $t \leftarrow t+1$
6) If the termination criteria are not met, go to (2)

In the end, the idea behind Bayesian optimization is to use the model through the Gaussian process in order to calculate the next best point to evaluate, which is calculated using the acquisition function. In this way, the problem becomes optimizing the acquisition function in each evaluation, the cost of which is considered negligible compared to evaluating in the objective function since it lacks noise and is explicitly counted, making it easier to optimize.

a) Acquisition function: The acquisition function is of great importance within Bayesian optimization since it regulates the trade-off between exploitation and exploration. There are multiple methods used, such as the probability of improvement (PI), expected improvement (EI), lower confidence bound (LCB), entropy search (ES), and a portfolio of several strategies, the latter usually giving better results. The values returned by the acquisition function usually correspond to the values expected to have a more significant improvement in the optimization task. The first of the listed strategies, probability of improvement, as the name suggests, measures the probability that the following observation is better than the best value obtained so far. Since the values of f (x) are those of a Gaussian posterior distribution, the mathematical expression that defines the probability of improvement is given by

$$PI(x) = P(f(x) \leq f(x^+)) - \varepsilon$$

$$= \Phi(\frac{f(x^+) + \varepsilon - \mu(x)}{\sigma(x)} \tag{2}$$

where $\phi(\cdot)$ is the cumulative probability function of standard Gaussian distribution, ε is a regularization constant, and $f(x^+)$ is the best value obtained up to that moment. The problem with this strategy is that it is pure exploitation; once there is a point with a good result, it directs the search only in the vicinity of that point, so it tends to stay local minimums. To also consider unexplored areas, the focus has to be changed to maximize the expected improvement. In order to achieve this, you must first define what the improvement is so that following, you obtain:

$$I(x) = \max\{0, f(x^+) - f(x_n + 1)\} \tag{3}$$

In which $I(x)$ will only have positive values when the evaluation at the new point $x_n + 1$ is less than the previous lower value $f(x^+)$. In order to continue with the calculation of the expected improvement, it must be taken into account that a Gaussian defined the values of $f(x)$, so the probability of improving $I(\cdot)$ in the posterior distribution defined by $\mu(x)$ and $\sigma^2(x)$, is obtained by solving the following integral.

$$\Xi(I) = \int_0^{\infty} \frac{1}{\sqrt{2\pi}\sigma(x)} \exp(-\frac{f(x^+) + \xi - \mu(x) - I^2}{2\sigma^2(x)})dI \quad (4)$$

where $I = I(x)$, gives as a result, the expected improvement.

$$\begin{cases} EI(x) = (f(x^+) + \xi - \mu(x)\Phi(Z) + \sigma(x)\phi(Z) & \text{if } \sigma(x) > 0 \\ 0 & \text{if } \sigma(x) = 0 \end{cases} \quad (5)$$

where

$$Z = \frac{f(x^+) + \xi - \mu(x)}{\sigma(x)} \quad (6)$$

Where $\mu(\cdot)$ is the probability density function of a standard Gaussian distribution.

In both equations shown above, the regulation constant ξ, which is added to $f(x^+)$, is used to specify that there could be a value higher than the best found.

Finally, we will discuss the acquisition function that uses the lowest confidence limit (or the highest, if it is being maximized). This function is optimistic about the variance $\sigma(x)$. Next, the mathematical expression of this strategy is presented

$$LCB(x) = \mu(x) - \nu\sigma(x) \quad (7)$$

where ν is a constant called kappa. Note that $v \geq 0$ must be met (since we want to minimize).

IV. Methodology

As seen in Fig. 3, the present work shows the architecture of the proposed MKCovid-19 workflow. It is made up of two main components of Training and Inference. Each element consists of several sub-components. The training component has a data separation structure, followed by data pre-processing and the sub-component of automatic hyperparameter optimization. Together, they make up the flow training and search for the best model considering the different configurations.

A. Training Component

The flow and procedures to train our MKCovid-19 model for the diagnosis of Covid-19 are detailed, which will depend on the order in which they are performed.

1) Split Dataset: MKCovid-19 uses 75% data for training and 25% for validation. The *validation data* is used to avoid overfitting the models. Besides, *test data* is also used to evaluate the model's performance in real situations, that is, the generalizability of the model.

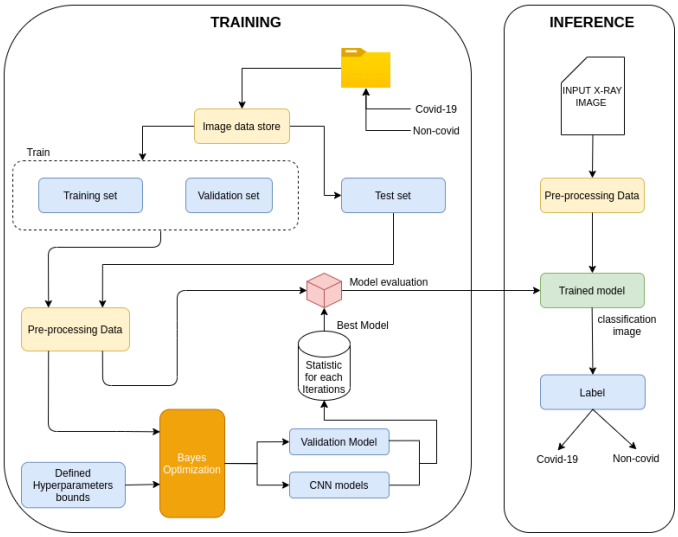


Fig. 3. Architecture of the Proposed MKcovid-19 Model.

2) Data Pre-Processing: The input data are prepared following the pre-processing pattern used in training the pre-trained models used in this work. Specifically, the images' size and normalization are converted to 224 * 224 pixels, being the standard size used by the popular models of convolutional neural networks (CNN).

3) Automatic Optimization of Hyperparameters: This sub-component receives a search space that contains the values that will not be optimized, such as iterations number and epoch number. It also receives all the hyperparameters that will be optimized, consisting of the learning rate, the batch size, momentum, pre-entered models, and gradient optimizers. At the end of each iteration executed, the statistical data is saved in a CSV file. These data are the values of each hyperparameter chosen by the Bayes optimizer, the loss of each model, and the iteration number. Subsequently, the best configuration is chosen based on the lowest loss obtained. With these configurations of the recovered hyperparameters, the final model is trained to evaluate it with the test data not seen by the training model.

B. Inference Component

The final model generated in the training component is used in this component. For any image consulted, it goes through the same data pre-processing. Finally, the model manages to classify the input image as Covid-19 or Non-Covid-19. This inference component can now be consumed and used by anyone, such as a medical specialist.

V. Experimental Setups

As mentioned in the previous sections, one of the main objectives is to provide information on which pre-trained model and hyperparameters can make it possible to obtain more efficient models in the classification of clinical images, specifically in detecting people with Covid-19. This section details the procedures, definitions, and metrics used to carry out our experiments to achieve this objective.

A. Separation of Data into Different Sets

To evaluate the precision of the resulting models trained using transfer learning for clinical data, specifically images of lungs compromised by Covid-19. We want to identify how much it affects the amount of training data, starting from small datasets, up to considerable amounts, in this case, 2,600 images. This experiment is evaluated with the validation dataset and the test dataset.

For this experiment, from the total of 2,600 selected datasets, datasets of different sizes were randomly drawn, such as 150, 300, 500, 1000, 1500, 2000, and 2600 (total data). The complete optimization procedure was executed for each data set, detailed in the Subsection 3.

B. Definition of the Optimization Space

A general non-optimized configuration was defined, with 50 epochs for each iteration and a total of 50 optimization iterations. The hyperparameters search space contains learning rate, momentum, pre-trained models, the gradient descent optimization function, and the batch size. We can observe these hyperparameters in Table I, where it is also shown for continuous data, the minimum, maximum value, and the increment. For discrete data, the set of values is shown.

TABLE I. DEFINE SEARCH SPACE FOR BAYES OPTIMIZATION

Hyperparameters	Min Value	Max Value	Step Value
Batch Size	5	100	5
Momentum	0	1	0.1
Learning Rate	log(0.01)	log(0.02)	-
Pre-Trained Models	Resnet18, Resnet50, Googlenet, Vgg16, Squeezenet, Densenet		-
Optimizer	SGD, Adam, RMSprop, Adagrad, Adadelta, Adamax		-

C. Evaluation Metrics

To evaluate the performance of the resulting model, we have used some metrics that recommend state of the art, such as Accuracy (Acc), precision (Pe), specificity (Sp), sensitivity (Se), and F- score. Besides, the count indices are reported, such as True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) that are used to calculate the metrics as mentioned above, which also allow us to have better analysis and understanding of them.

1) Accuracy (Acc): $\frac{t_p+t_n}{t_p+t_n+f_p+f_n}$

2) Precision (Prc): $\frac{t_p}{t_p+f_p}$

3) Specificity (Spc): $\frac{t_n}{t_n+f_p}$

4) Sensitivity (Sen): $\frac{t_p}{t_p+f_n}$

5) F1-Score (F1): $\frac{2t_p}{2t_p+f_p+f_n}$

D. Implementation Details

To implement and evaluate our proposed model, Google Colab [23] was used, which provides an Intel (R) Xeon (R) CPU @ 2.30 GHz, a 12 GB system memory, and a 16 GB Tesla P100 GPU. The implementation of this model was done in the Python 3.5 [2] language. We used the pre-trained models available from the Pytorch library [24]. For the Bayesian optimization of hyperparameters, the Hyperopt library[25] was used explicitly to optimize minimization. All the data used and the codes developed for this work are available for future study in the repository[3].

VI. EXPERIMENTAL RESULTS

In this section, an analysis is made of the hyperparameters' values obtained throughout the Bayesian optimization iterations. The frequency of use of each hyperparameter is analyzed together with their selection over time. Finally, the results are presented based on the metrics mentioned in Subsection V-C, comparing the models obtained with the validation data and test data.

A. Analysis of the Use of Pre-trained Models and Gradient Optimizers

In Fig. 4, the number of times that the optimizer used a pre-trained model is observed. The frequent use could indicate that a better model for classification has been obtained using that pre-entered model. The result of executing the optimization for each of the datasets shows us that each one chose different pre-trained models, specifically for the data sets considered small, such as 150, 300, 500, and 1000. This behavior can be explained that when trying to minimize the loss in each iteration by choosing a pre-trained model, overfitting occurred due to the small training data.

On the contrary, if we analyze the frequency of the data sets with more data in this work, that is, 1500, 2000, and 2600 (totality), we can observe that the Densenet model is used more frequently. However, the set of 2600 data also has a similar high usage frequency to the Squeezenet model. These results could indicate that our Bayes optimizer could have decided and/or modeled that Densenet is the best pre-trained model for this problem. Furthermore, we conclude that the Vgg16 model is not recommended for this type of application.

Similarly, in Fig. 5, it is shown that the best optimizer for gradient descent for almost all data sets was Adamax. The difference in frequency of use compared to the other optimizers is considerable. This statistic seems to verify the theory and recommendation on using this optimizer by state of the art.

B. Analysis of the Behavior of the Hyperparameter Distributions

Fig. 6 show the sampling distributions and the search choice of the best hyperparameters for learning rate, batch size, and momentum for each dataset. It is observed that for the Learning Rate, the curves have a similar shape and similar values, independent of each data set's data size. These curves

[2]https://www.python.org/

[3]https://github.com/kvvaldez/MKCovid-19

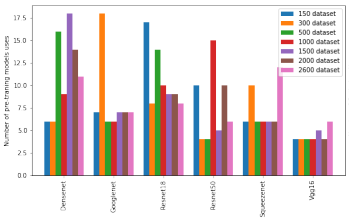


Fig. 4. figure
Frequency of use of Pre-trained Models During Bayesian Optimization.

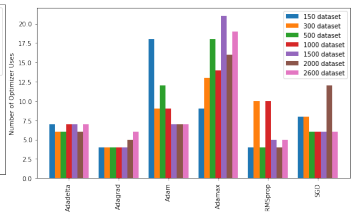


Fig. 5. figure
Frequency of use of Gradient Optimizers During Bayesian Optimization.

have a similar shape to the probability sampling curve but with a range variation of the x-axis. This hyperparameter's best values in the present work range from $< 0.01, 0.035 >$. This range of values could help us define the initial values for a better fit and explore the smallest search space.

For the batch size curve, it is shown that the curves of different data sizes differ from probabilistic sampling, but these curves coincide that it is advisable to use small batch size data in the range of $< 8, 18 >$.

Also, it should be remembered for the momentum curve that this hyperparameter is only used by the SGD and RMSprop gradient optimizers. The momentum figure shows two ridges in the curves in the experiments: $< 0, 0.03 >$ and the other $< 0.8, 0.99 >$, which is close to one. It is necessary to consider that hyperparameter ranges found during Bayesian Optimization are not necessarily better for the test set, only that they produce less loss in the validation data.

Finally, Fig. 7 and Fig. 8 show how the use of pre-trained models and gradient optimizers change over time throughout Bayes optimization. The figure is shown all iterations for each dataset, where it can be observed before iteration 20, many elements belonging to our search space are tested. Later, starting at iteration 20, the optimizer takes a limited number of specific gradient optimizers and pre-trained models. Similarly, in Fig. 10, for each dataset, accuracy was maximized. Although it is observed that the optimization has been minimal, this happens because good results are obtained naturally, at least for this problem. This difference of 1 or two points in the accuracy can mean much gain when using these models to use covid-19 diagnosis and consequently save lives.

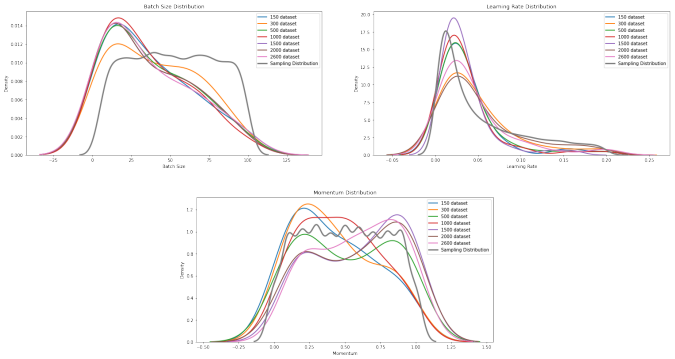


Fig. 6. Sampling Distribution and the Distribution of Learning Rate, Momentum, and Batch Size for the Different Dataset Sizes.

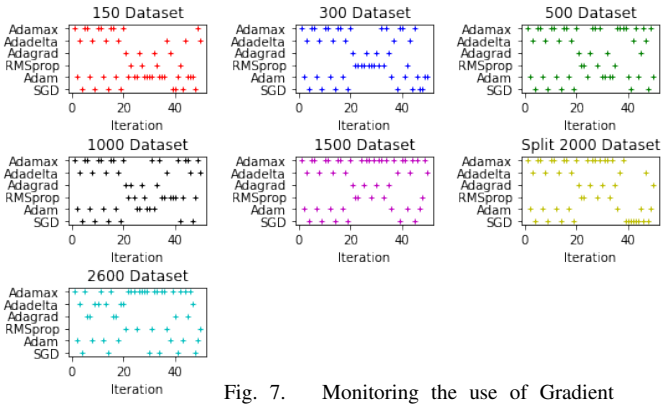


Fig. 7. Monitoring the use of Gradient Optimizer Throughout the Optimization.

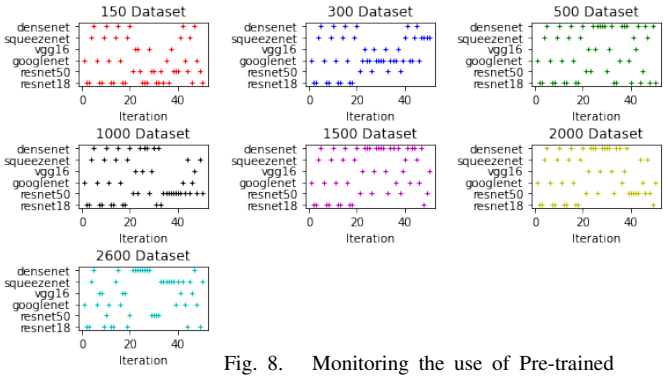


Fig. 8. Monitoring the use of Pre-trained Models Throughout the Optimization.

C. Classification Results and Discussion

Fig. 9 shows the best models' validation precision for each data set. To obtain the best model, all the hyperparameter values were ordered in increasing order according to the loss obtained in each iteration. Then the first best hyperparameter configuration was extracted for each data set.

It is also observed in Fig. 9 the obtaining of models above 94% precision. It seems to confirm that using transfer learning works well for any data amount. Also, from the data set 1000 onwards, a very similar precision was obtained, which is above 97%. The models' performance validation throughout the optimization was carried out with data that simulates a real-world situation. That is, more people think they have Covid-19 without having it, compared to people who have it. This is because their symptoms are not only symptoms of a single disease. For this simulation, the test data was not seen in the training. This data consists of 183 images tagged with Covid-19 and 500 images of people without Covid-19.

The result obtained with the test data is shown in Table II, using a threshold of 0.4. It is observed that low accuracy was obtained for '150 dataset' because there are many false negatives. On the contrary, for '300 dataset', an accuracy of 88% was obtained, which could be considered a good value, but compared to 98% in the validation, it is a considerable difference. This difference in accuracy values could verify the hypothesis mentioned within the Subsection VI-A where the model cannot learn or generalize its learning because little data was used for training. From the "500 dataset" data set, accuracy values similar to the validation one are obtained,

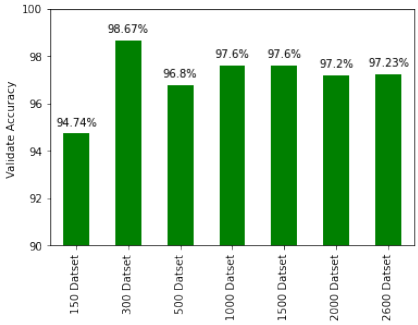


Fig. 9. Validate Accuracy of the Best Models from Each Data Set.

obtaining the best accuracy for data totality (2600 dataset). This high accuracy value is due to the fact that the model was trained with a more significant amount of data, which manages to learn and generalize all the characteristics of an image with a diagnosis of Covid-19.

TABLE II. RESULTS FOR TEST DATASET, BASED ON DIFFERENT METRICS.

DATASET	FN	FP	TN	TP	F1	ACC	PRC	SEN	SPC
150 dataset	223	3	277	180	0.614	0.67	0.98	0.98	0.55
300 dataset	59	23	441	160	0.80	0.88	0.87	0.87	0.88
500 dataset	41	10	459	173	0.87	0.93	0.95	0.95	0.92
1000 dataset	10	12	490	171	0.94	0.97	0.93	0.93	0.98
1500 dataset	11	16	489	167	0.93	0.96	0.91	0.91	0.98
2000 dataset	6	13	494	170	0.95	0.97	0.93	0.93	0.99
2500 dataset	11	5	489	178	0.96	0.98	0.97	0.97	0.98

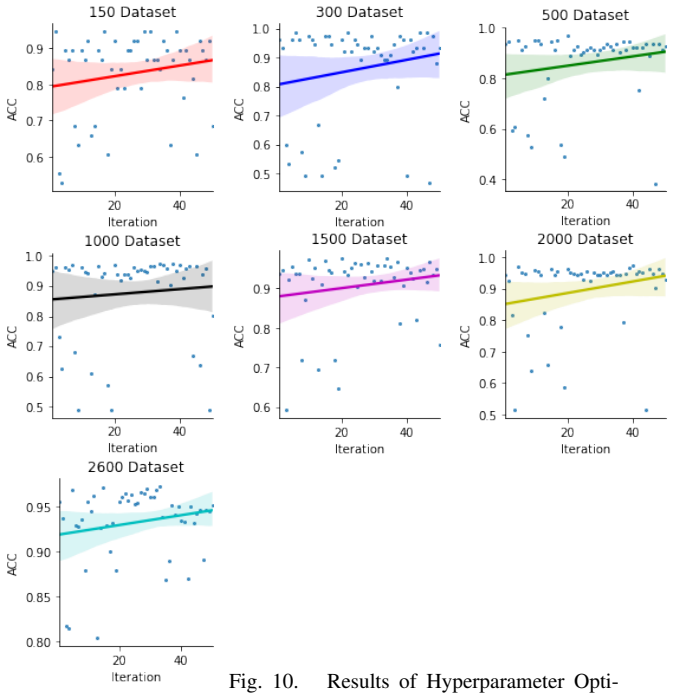


Fig. 10. Results of Hyperparameter Optimization across Iterations.

VII. CONCLUSION

A detailed study was conducted, and a workflow was presented for the automatic selection of hyperparameters for the classification of images of people with Covid-19 using the Pytorch library. Several experiments were carried out, within which the hyperparameters were optimized for each data set with different amounts of data. The pre-trained models were included as a hyperparameter in order to try to know and recommend which models best fit this type of medical image classification model. For this reason, we conclude the best fit of the pre-entered models towards our problem is highly variable according to the amount of data. If we consider the results obtained with relatively large data sets, the Densenet models and Resnet in general, provide us with accurate and robust models. As a result of this study, we made available the best hyperparameters of the best model obtained with an accuracy of 98%, which will help to quickly detect people with Covid-19 with high sensitivity, using chest X-ray images. Besides, due to the analysis seen in the VI-B sections, we recommend smaller search spaces for this problem, which will surely be very useful as a starting point for people who want to train a model similar to ours. We then conclude that Bayesian optimization is an effective strategy to increase transfer learning use cases.

VIII. FUTURE WORKS

As a first future work, we recommend doing more iteration in the optimization component to analyze better and conclude the behavior of the hyperparameters considered in the search space in this work. We recommend testing the other pre-trained models not addressed in this study regardless of the image size they were trained in since only models with image size 224 * 224 pixels were used. Finally, it is recommended to validate this optimization workflow to classify other diseases that are not necessarily Covid-19 images.

REFERENCES

[1] Majid Nour, Zafer Cömert, and Kemal Polat. A novel medical diagnosis model for covid-19 infection detection based on deep features and bayesian optimization. *Applied Soft Computing*, 97:106580, 2020.

[2] Linda Wang, Zhong Qiu Lin, and Alexander Wong. Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. *Scientific Reports*, 10(1):1–12, 2020.

[3] Mohammad Rahimzadeh and Abolfazl Attar. A modified deep convolutional neural network for detecting covid-19 and pneumonia from chest x-ray images based on the concatenation of xception and resnet50v2. *Informatics in Medicine Unlocked*, 19:100360, 2020.

[4] Antonios Makris, Ioannis Kontopoulos, and Konstantinos Tserpes. Covid-19 detection from chest x-ray images using deep learning and convolutional neural networks. In *11th Hellenic Conference on Artificial Intelligence*, pages 60–66, 2020.

[5] Konstantin Pogorelov, Michael Riegler, Sigrun Losada Eskeland, Thomas de Lange, Dag Johansen, Carsten Griwodz, Peter Thelin Schmidt, and Pål Halvorsen. Efficient disease detection in gastrointestinal videos–global features versus neural networks. *Multimedia Tools and Applications*, 76(21):22493–22525, 2017.

[6] Konstantin Pogorelov, Kristin Ranheim Randel, Carsten Griwodz, Sigrun Losada Eskeland, Thomas de Lange, Dag Johansen, Concetto Spampinato, Duc-Tien Dang-Nguyen, Mathias Lux, Peter Thelin Schmidt, et al. Kvasir: A multi-class image dataset for computer aided gastrointestinal disease detection. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, pages 164–169, 2017.

[7] Michael Riegler, Mathias Lux, Carsten Griwodz, Concetto Spampinato, Thomas de Lange, Sigrun L Eskeland, Konstantin Pogorelov, Wallapak Tavanapong, Peter T Schmidt, Cathal Gurrin, et al. Multimedia and medicine: Teammates for better disease detection and survival. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 968–977, 2016.

[8] Qiao Ke, Jiangshe Zhang, Wei Wei, Dawid Połap, Marcin Wo'zniak, Leon Ko'smider, and Robertas Damasevcius. A neuro-heuristic approach for recognition of lung diseases from x-ray images. *Expert Systems with Applications*, 126:218–232, 2019.

[9] Xianghong Gu, Liyan Pan, Huiying Liang, and Ran Yang. Classification of bacterial and viral childhood pneumonia using deep learning in chest radiography. In *Proceedings of the 3rd International Conference on Multimedia and Image Processing*, pages 88–93, 2018.

[10] Matteo Polsinelli, Luigi Cinque, and Giuseppe Placidi. A light cnn for detecting covid-19 from ct scans of the chest. *Pattern Recognition Letters*, 140:95–100, 2020.

[11] Sakshi Ahuja, Bijaya Ketan Panigrahi, Nilanjan Dey, Venkatesan Rajinikanth, and Tapan Kumar Gandhi. Deep transfer learning-based automated detection of covid-19 from lung ct scan slices. *Applied Intelligence*, 51(1):571–585, 2021.

[12] Saman Motamed, Patrik Rogalla, and Farzad Khalvati. Randgan: Randomized generative adversarial network for detection of covid-19 in chest x-ray. *arXiv preprint arXiv:2010.06418*, 2020.

[13] Chris Thornton, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 847–855, 2013.

[14] Haifeng Jin, Qingquan Song, and Xia Hu. Auto-keras: An efficient neural architecture search system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1946–1956, 2019.

[15] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D Sculley. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1487–1495, 2017.

[16] Rune Johan Borgli, Håkon Kvale Stensland, Michael Alexander Riegler, and Pål Halvorsen. Automatic hyperparameter optimization for transfer learning on medical image datasets using bayesian optimization. In *2019 13th International Symposium on Medical Information and Communication Technology (ISMICT)*, pages 1–6. IEEE, 2019.

[17] Ibrahem Kandel and Mauro Castelli. The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. *ICT express*, 6(4):312–315, 2020.

[18] M. E. H. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M. A. Kadir, Z. B. Mahbub, K. R. Islam, M. S. Khan, A. Iqbal, N. A. Emadi, M. B. I. Reaz, and M. T. Islam. Can ai help in screening viral and covid-19 pneumonia? *IEEE Access*, 8:132665–132676, 2020.

[19] Tawsifur Rahman, Amith Khandakar, Yazan Qiblawey, Anas Tahir, Serkan Kiranyaz, Saad Bin Abul Kashem, Mohammad Tariqul Islam, Somaya Al Maadeed, Susu M. Zughaier, Muhammad Salman Khan, and Muhammad E.H. Chowdhury. Exploring the effect of image enhancement techniques on covid-19 detection using chest x-ray images. *Computers in Biology and Medicine*, 132:104319, 2021.

[20] Amith Khandakar Tawsifur Rahman, Dr. Muhammad Chowdhury. Covid-19 radiography database, 2021. https://www.kaggle.com/tawsifurrahman/covid19-radiography-database.

[21] Shervin Minaee, Rahele Kafieh, Milan Sonka, Shakib Yazdani, and Ghazaleh Jamalipour Soufi. Deep-covid: Predicting covid-19 from chest x-ray images using deep transfer learning. *arXiv preprint arXiv:2004.09363*, 2020.

[22] Daniel Fernández Sánchez. Creating a bayesian optimization tool in python. pages 15–18, 2019.

[23] Ekaba Bisong. Google colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 59–64. Springer, 2019.

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[25] James Bergstra, Dan Yamins, David D Cox, et al. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *Proceedings of the 12th Python in science conference*, volume 13, page 20. Citeseer, 2013.