information

MDPI

Article

Convolutional Support Vector Models: Prediction of Coronavirus Disease Using Chest X-rays

Mateus Maia [1], Jonatha S. Pimentel [2], Ivalbert S. Pereira [2], João Gondim [3], Marcos E. Barreto [3] and Anderson Ara [2,*]

[1] Hamilton Institute, Mathematics and Statistics, Maynooth University, County Kildare, W23 F2K8 Maynooth, Ireland; mateus.maiamarques.2021@mumail.ie

[2] Statistics Department, Federal University of Bahia, Salvador 40170-110, Brazil; jsppimentel9@gmail.com (J.S.P.); ivalbert.pereira@gmail.com (I.S.P.)

[3] Computer Science Department, Federal University of Bahia, Salvador 40170-110, Brazil; gondimjoaom@gmail.com (J.G.); marcosb@ufba.br (M.E.B.)

* Correspondence: anderson.ara@ufba.br

Received: 3 September 2020; Accepted: 19 November 2020; Published: 26 November 2020

check for updates

Abstract: The disease caused by the new coronavirus (COVID-19) has been plaguing the world for months and the number of cases are growing more rapidly as the days go by. Therefore, finding a way to identify who has the causative virus is impressive, in order to find a way to stop its proliferation. In this paper, a complete and applied study of convolutional support machines will be presented to classify patients infected with COVID-19 using X-ray data and comparing them with traditional convolutional neural network (CNN). Based on the fitted models, it was possible to observe that the convolutional support vector machine with the polynomial kernel ($CSVM_{Pol}$) has a better predictive performance. In addition to the results obtained based on real images, the behavior of the models studied was observed through simulated images, where it was possible to observe the advantages of support vector machine (SVM) models.

Keywords: COVID-19; X-ray; CNN; SVM; convolution

1. Introduction

The coronavirus disease (COVID-19) is an ongoing pandemic that spread quickly worldwide. The first case was noted in Wuhan City, China, in November 2019. Later, a high number of cases were reported in diverse countries making the World Health Organization (WHO) announce the pandemic as a public health emergency on 11 March 2020. WHO also announced that the virus can cause a respiratory disease with a clinical presentation of cough, fever, and lung inflammation. Another aggravating factor is that the virus has a high spreading ratio for person-to-person contamination [1], which puts self-isolation measures and urgent tracking and diagnosis of possible cases as priorities to stop virus propagation.

Several works were published to track the spreading of the disease [2,3] and to establish a clear comprehension of the state of the pandemic around the world [4,5]. Some works showed that one of the diagnosis methods is handled by radiologists, who performs a manual lung infection quantification caused by the virus [6]. Reference [7] showed that as the number of patients infected increases, it turns out to be more difficult for radiologists to timely finish the diagnosis. Therefore, statistical learning models can be extremely useful to embrace a greater number of cases to provide accurate and faster support for diagnosing COVID-19 and its complications.

Data-driven solutions proposing automatic diagnosis using statistical learning methods applied to medical images are present in the literature [8–12]. Recently, convolutional neural networks

(CNNs) have been recognized as a powerful tool for predictive tasks using medical images [13–15]. The support vector machine (SVM) [16] is a model that can achieve great prediction and generalization capacity, due to its theoretical properties. These characteristics are supported by a convex optimization, based on the structural risk minimization principle, to obtain the minimum global of a loss function. This principle distinguishes SVM from other learning algorithms because it guarantees a global minimum and avoids being trapped in a local minimum as perceived in neural networks.

SVM also presents a notable performance applied in image classification tasks. Several works used support vector models for image recognition with great observed performance [17–22]. More recently, with the advent of convolutional neural networks to classify images, SVM was applied over convolutional features and achieved good results, presenting a general error lower than the traditional CNN approach [23–26].

The use of support vector machines applied to medical images has been an active research field [27–30]. Data-driven diagnosis of COVID-19 through X-ray images using the convolutional support vector model (CSVM) have shown the great algorithm's predictive capacity of identifying patients stricken by the disease [31,32].

The contribution of this paper relies on how the convolutional support vector machine can be used in X-ray images to identify COVID-19 cases. Recently several models of automated diagnosis of this disease were being presented by the scientific community [33]. Chen et. al [34] presented the mean average of 69 proposed models and in comparison with them we can see that the convolutional support vector approach produced more accurate results than most of them. In addition, we present a new and robust analysis through simulation studies and a consistent validation procedure based on hold out repetitions, exploring different kernel functions and architectures. In comparison with other deep learning procedures [35,36], and transfer learning [37], this paper presents a model with higher accuracy, overcoming them by 2%, on average. Sophisticated models that applied the pre-processing algorithms jointly with a deep-learning framework [38] were not able to generate predictions that outperformed the CSVM proposed in this paper.

The remaining sections of this paper are organized as follows: Section 2 exposes the methodology of the convolutional network using the traditional deep learning approach. Section 3 exposes the support vector machine methodology and the convolutional support vector models. Section 4 discusses the simulation study. Section 5 displays the results and discussion. Finally, Section 6 closes the paper with the final considerations.

2. Convolutional Networks

Convolutional neural networks (CNN) are multi-layer architectures where the successive layers are designed to progressively learn high-level features, being the last layer responsible for producing a result [22]. They have been shown to be extremely accurate for time series analysis and image classification [39,40]. The convolutional layers present in CNN are responsible for applying filters throughout the image and thus reduce its complexity. When an image is filtered, the output is another image that can be understood as a feature map indicating whether certain features (for example, borders) were detected in the input image [41].

The process of a convolutional network happens as depicted in Figure 1: first, an input image is convoluted with filters, thus reducing its size. Then, a new matrix will show the results of the convolution operations. These results are grouped through a pooling operation, which extracts the maximum values from small regions of interest. Finally, the grouped matrix is decimated by a factor of two to produce the final result, through the decimation operation. The result is conducted by a classification method responsible to predict the label which summarizes the main content detected on the image.

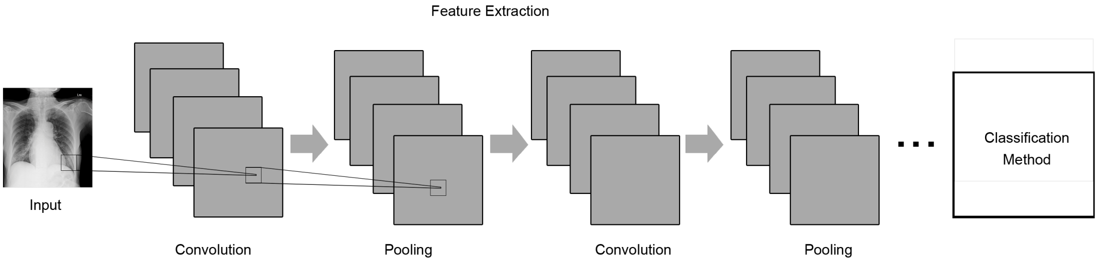


Figure 1. Generic model architecture with convolutional layers. Source: prepared by the authors and inspired by [42].

2.1. Multi-Layer Perceptron Neural Networks

Neurons are the main cells that make up the nervous system, responsible for conducting, receiving, and transmitting nerve impulses throughout the body as a response from stimuli from the environment, for example. The brain is a complex network that processes information through a system of several interconnected neurons. It has always been challenging to understand brain functions; however, due to advances in computing technologies, it is possible to program artificial neural networks capable of mimicing some known brain capabilities [43]. Perceptrons are the basic building blocks of artificial neural networks. A perceptron implements a linear equation that takes as input all the relevant variables (X_i) and their associated weights (w_i), performs the calculation, and generates a binary output (0 or 1) by comparing the result with a given threshold. Figure 2 shows a basic representation of a perceptron.

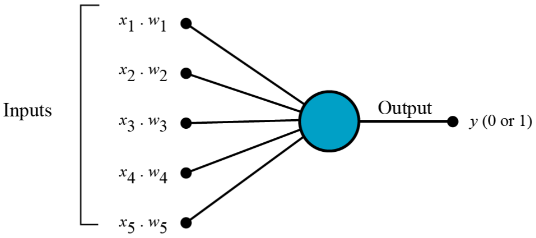


Figure 2. Basic representation of a perceptron. Source: [44].

A multi-layer perceptron network can be built by a sequence of layers, as depicted in Figure 3. The first layer takes decisions based on the input variables and their weights; then perceptrons in the subsequent (hidden) layers can make more complex decisions by weighing the results from the previous layers. The output layer is responsible to generate a result (usually, a classification) based on some specific decision function.

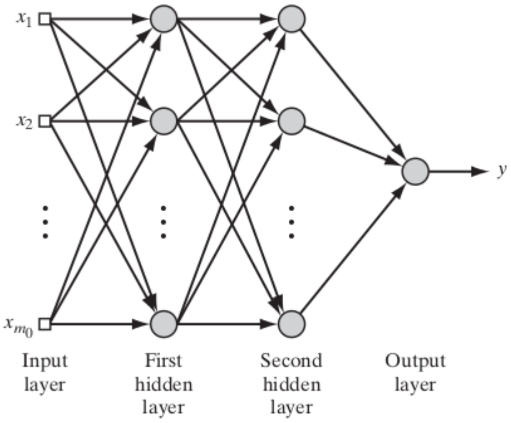


Figure 3. MLP architecture. Source: [44].

2.2. Convolution Neural Networks

The so-called convolutional neural networks (CNNs) can be seen as an extension of common neural networks, in which new pre-processing layers can be added to extract patterns during image analysis, for example. Usually, after the application of the convolutional layers, a neural network model is applied. However, nothing prevents the use of other models at this step. Some studies, such as [45,46], followed this approach, using an SVM model at the end of the convolutional layers instead of the commonly used neural network.

The layers used are a set of filters that are convoluted in order to generate a set of features map. Among the convolutional layers used, there is a pooling layer, which operates over blocks of the input set and combines the activation features. This combining operation is defined by a grouping function, such as average or maximum. This pooling process step is used to summarize the nearby outputs statistically according to the pooling function [40] creating patches of feature maps. Since pooling downsizes the output, another step of pre-processing the image is zero padding, which allows us to control the size of the output [40]. Padding is the process of adding pixels to the border of the image. After this feature extraction step, this output is sent to a fully connected layer. In general, a traditional *CNN* uses dense multi-layer perceptrons as the classification method, as pointed out in Figure 4.

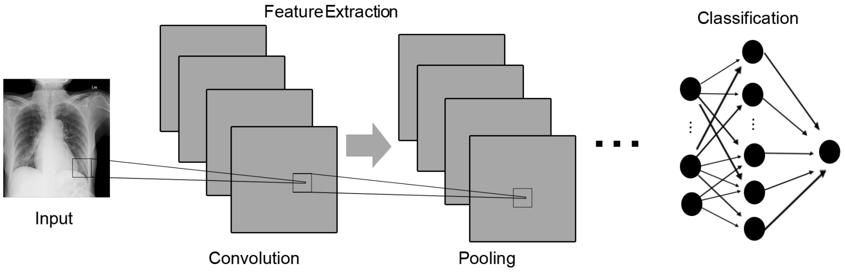


Figure 4. Convolutional neural network (CNN) architecture. Source: prepared by the authors and inspired by [42].

3. Support Vector Machines

Cortes and Vapnik [16] proposed a new class of models known as support vector machines (SVM), which are based on the theory of Statistical Learning [47]. Basically, this class of models is based on the construction of optimal hyperplanes to classify labels—usually into two distinct classes—as well as on the use of kernelization to increase the model's flexibility [48].

SVM is used to solve classification problems, finding the hyperplane with the greatest separation space (delimited by its margins) between the categories of the chosen variable. Figure 5 displays the general representation of an SVM. The hyperplane is the equation separating the two classes, while the support vectors are the data points closest to the margin boundaries.

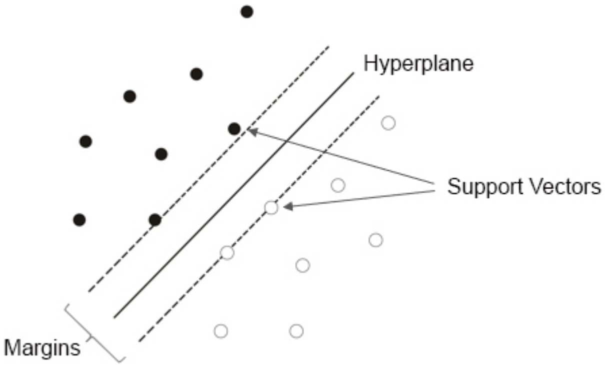


Figure 5. Support vector machine (SVM) representation showing the hyperplane, margins, and associated support vectors. Source: prepared by the authors.

3.1. Support Vector Classifier

The simplest form of SVM is called support vector classifier (or linear SVM), which is a technique that allows for linear separation of data whenever possible. This technique can be used with rigid (hard) margins, which do not allow for misclassification but hinders generalization; or soft margins, which allow the entry of wrongly classified data but improve the model generalization. Figure 6a depicts an SVM using rigid margins, with no data points between the margins; while Figure 6b depicts a soft margin SVM.

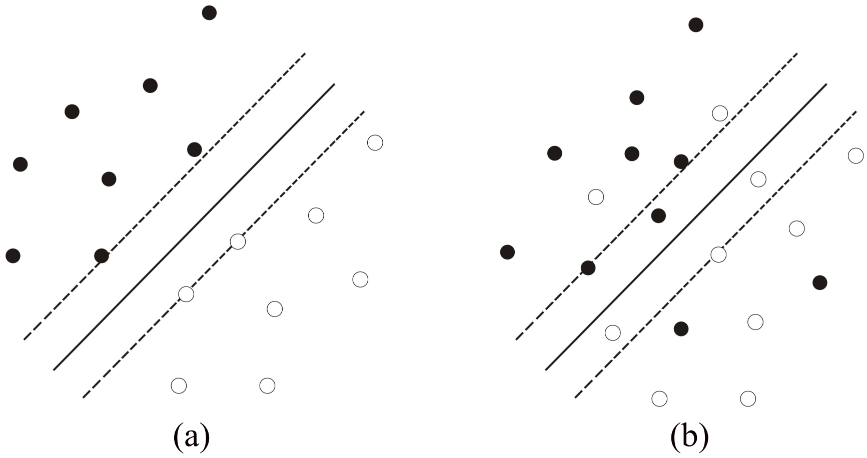


Figure 6. Representation of a linear SVM using (**a**) rigid margins and (**b**) soft margins. Source: Adapted from [49] and prepared by the authors.

In SVM with soft margins, with n the sample size of training data and $i = 1, \ldots, n$, the ε_i are slack variables (errors) which are added to relax the restrictions and turn the model more receptive to new data. Therefore, the function to be maximized or minimized can be expressed as:

$$\min \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \varepsilon_i, \tag{1}$$

since C is a regularization constant that imposes a weight to minimize errors, there is no limit to the number of wrong classifications. If $C \to \infty$, a soft margin SVM will act as a rigid margin SVM. After applying the maximization process via Lagrange multipliers of the Equation (1) with margins constraints, the SVM classifier with soft margins is given by:

$$f(x) = sign\left(\sum_{i=1}^{n} \alpha_i y_i (x_i \cdot x) + b\right). \tag{2}$$

where b is a constant parameter also called bias. The α_i are the Lagrangian parameters that identify the support vectors when $\alpha_i > 0$ implies that x_i is a support vector and an important observation in the entire model. The function $sign$ is responsible to verify if any observation is upper or lower of the estimated hyperplane.

3.2. Kernel Transformation

In simplest situations where the data can be easily classified through a linear equation, a linear classifier solves the problem with good levels of accuracy. However, in many situations, this linear classifier will encounter problems to separate the data and require the use of a non-linear classifier. This nonlinear classifier via kernelization procedure maps the training set from its original space, referred to as (R) input space, into a new, larger space, called (F) characteristic space, as shown in Figure 7.

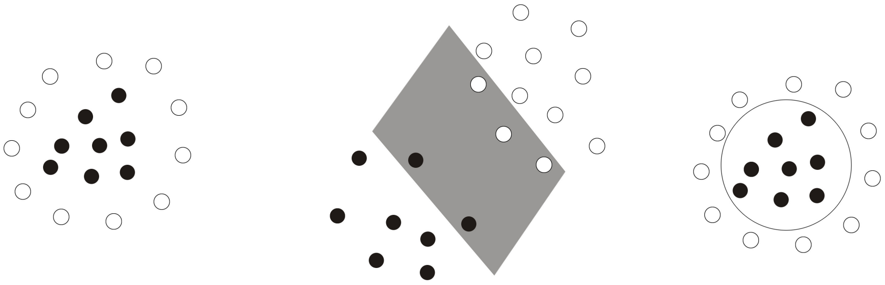


Figure 7. The kernel transformation (trick) process. A non-linear data space (left) is mapped, through a non-linear kernel, into a new linear separable data space (right). Source: Adapted from [49] and prepared by the authors.

This mapping is defined as a ϕ function. The appropriate choice of the ϕ mapping function means that the training set mapped into (F) can be now separated by a linear SVM. Thus, our classifier will be rewritten as:

$$f(x) = sign\left(\sum_{i=1}^{n} \alpha_i y_i (\phi(x_i) \cdot \phi(x)) + b\right) \tag{3}$$

since the kernel function $K(x, x') = \phi(x_i) \cdot \phi(x_i)$, Equation (3) differs from Equation (1) by adding the kernelization function, expressed by the aggregation of the map functions. In general, the Gaussian kernelization has a high predictive capacity and is being frequently used [48]. The most commonly used kernels [50,51] are shown in Table 1.

Table 1. Most used kernels for kernel transformation.

Kernel Type	$K(x, x')$	**Parameters**
Linear	$\gamma(x \cdot x') + d$	γ, d
Polynomial	$(\gamma(x \cdot x') + d)^q$	γ, d, q
Gaussian	$exp(\frac{-\|\|x-x'\|\|^2}{2\gamma^2})$	γ

3.3. Convolution Procedures and SVM

Analogously to a convolutional neural network (CNN), convolutional support vector machine uses the convolutional image processing to reduce the complexity of the image, through operations as the convolution and pooling as shown in Section 2.2. Initially a convolutional model is generated to select and optimize the filter's that will be used. Afterwards, with fixed hyper-parameters of the convolutional layers, the traditional support vector machine (SVM) is employed, and the model can be applied using any of the kernels discussed above, with their due parameters that can be estimated through a tuning process. Figure 8 shows the convolution support vector machine (CSVM) architecture.

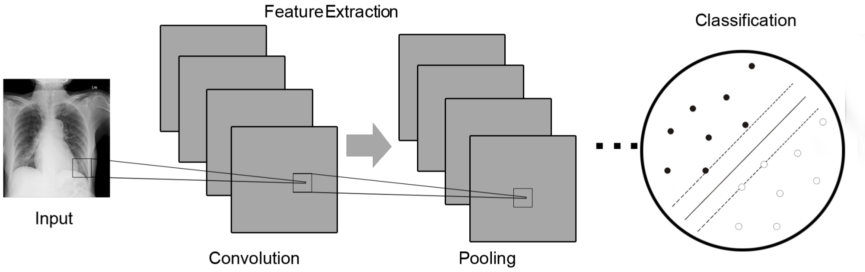


Figure 8. Convolution support vector machine (CSVM) architecture. Source: prepared by the authors and inspired by [42].

Among the existing layers in the pre-processing of data through convolution, we can mention two that have extremely important roles, pooling layer and flattening layer. Directly, the pooling layer seeks to somehow reduce the size of the data, so that information is not lost, but to keep those that have a greater amount of significant information, reducing noise in order to reduce processing time. In the final stage of the pre-processing stage, the flattening layer seeks to convert the data into a single observation in order to facilitate insertion into the model sequence. In general, the convolution acts as a complex processing filter which can be applied in other model instead of the classic Multilayer Perceptron (MLP). After the flattening step, SVM replaces MLP. Figure 9 shows flattening layer after convolution filter processing.

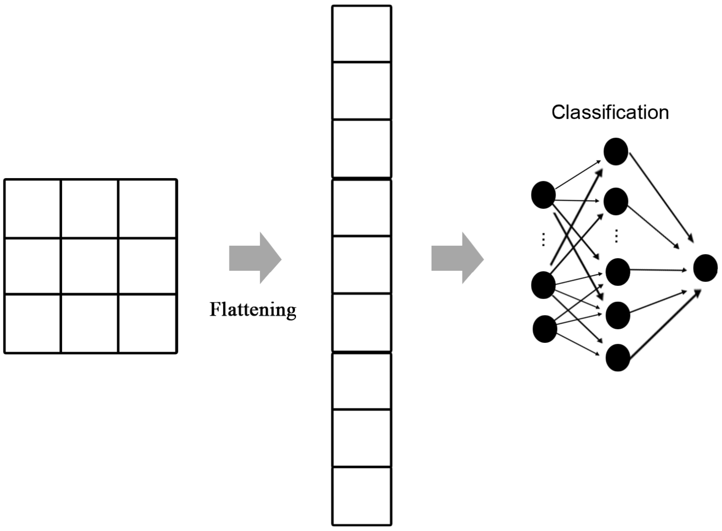


Figure 9. Representation of a flattening layer. Source: adapted from [52] and prepared by the authors.

4. Simulation Study

For this simulation, we have created synthetic 64 × 64 pixels images with three color channels (RGB) to compose two distinct classes. We aimed to evaluate the performance of different models, in terms of computation time and predictive performance. Each image class was created from a normal distribution with different means μ in the Green (G) channel, which produced two different validation sets per class. This is a similar artificial image procedure as in [53].

Furthermore, Table 2 presents the parameters used to generate the simulation sets. The first column indicates the mean difference between classes in standard deviation. Thus, 1SD means the difference in channel G is one standard deviation, while 3SD means the difference in channel G is equal to three standard deviations. In summary, there are four different kinds of images: Class1, with means equal to 119 and 101 in channel G; and Class2, with means equal to 137 and 155 in channel G. The other color channels are identical for all images. Figure 10 shows the artificial images used in our simulation.

Table 2. Parameters of each color channel per class.

		Class 1	**Class 2**	
μ Difference	**Channel**	μ		σ
1SD	R	128	128	25
	G	119	137	18
	B	128	128	30
3SD	R	128	128	25
	G	101	155	18
	B	128	128	30

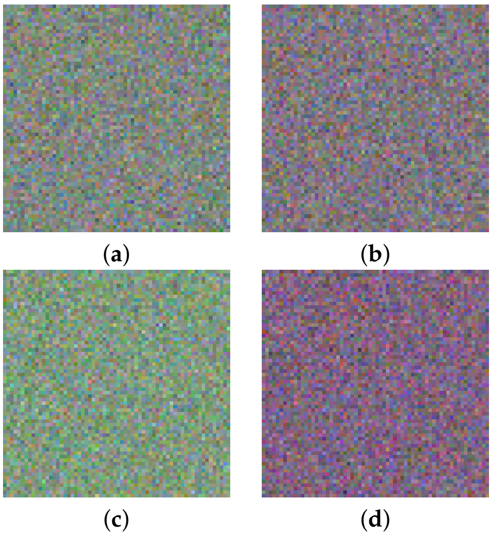


Figure 10. Synthetic 64 × 64 pixels images: (**a**,**b**) with μ difference in channel G equal to one standard deviation for classes 1 and 2; (**c**,**d**) with μ difference in channel G equals to three standard deviations for Classes 1 and 2. Source: prepared by the authors.

It is important to notice that the classification task can be easier if there is a higher means difference among the images. The next subsection details the model configuration.

Experimental Setup

In addition, we have used a CNN (called CNN_1) with the following filter configuration: first, a 3 × 3 convolutional layer with 30 filters, then a 2 × 2 pooling layer. Another CNN (called CNN_2) was used with a different initial filter configuration: first, a 3 × 3 convolutional layer with 30 filters, then a second 3 × 3 convolutional layer with 60 filters. Then we used a 2 × 2 pooling layer, plus a 3 × 3 convolutional layer with 60 filters, and finally a new 2 × 2 pooling layer.

In the convolutional neural network models, the two architectures were used for comparison purposes. For the first architecture (CNN_1), we used an initial layer with 256 nodes, a second layer with 128 nodes, the third layer with 64 nodes, and an output layer with 2 nodes. In the second architecture (CNN_2), we used an initial layer with 128 nodes, a second layer with 64 nodes, the third layer with 32 nodes, and finally a layer with 2 nodes. In addition, the activation function used by the CNN model was the *relu* function, as well as the adam optimizer, which sets by the default *learning rate* value to 0.001. A total of 420 images (210 with covid and 210 without covid) were used, requiring a total of 25 epochs to adjust the model. Another important parameters to be mentioned are the *batch size* with default value equals to 32, the *objective function*, adjusted to *sparse categorical crossentropy* and *accuracy* as metrics. Figure 4 presents a complete view of the CNN architecture used in our simulation.

At the convolutional SVM modeling, pre-trained weights from CNN are used as filters for the classification, being it is fixed in its training. Afterwards, the support vector component is adjusted with different kernel functions CSVM$_{RBF}$, CSVM$_{Lin}$, and CSVM$_{Pol}$, respectively, with their respective default parameters. The applied convolution filters were the same as the ones in CNN$_1$. Representations of the model setups can be seen in Figure 11.

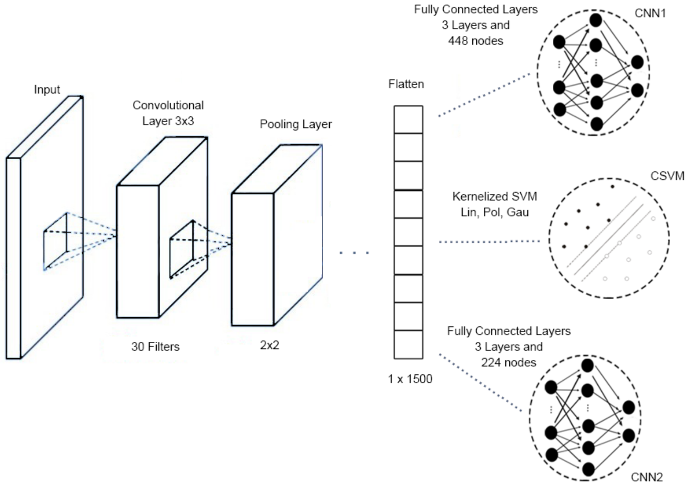

Input
Convolutional Layer 3x3
30 Filters
Pooling Layer
2x2
Flatten
1 x 1500
Fully Connected Layers 3 Layers and 448 nodes
CNN1
Kernelized SVM Lin, Pol, Gau
CSVM
Fully Connected Layers 3 Layers and 224 nodes
CNN2

Figure 11. Representation of models. Source: adapted from [54] and prepared by the authors.

Simulations were split into image batches of 100, 300, 500, and 1000 sample sizes with an equal balance of each class. First, for the image group with one standard deviation difference in means (named 1SD) and then for the image group with three standard deviation difference in means (named 3SD). We applied a training-test split in a proportion of 90/10 for a hundred cross validation repetitions.

As the simulation is a binary classification problem, it produces the well-known confusion matrix which correlates the predictions and true class values. The evaluation is made by the following metrics, calculated from the confusion matrix entries (Baldi et al. [55]):

- Accuracy (ACC): is the rate between correct predictions and total of predictions. It is sensitive to classes unbalancing.
- Sensitivity (SEN): known as recall, it is the rate of true positive predictions and all positive predictions.
- Specificity (SPC): it is the rate of true negative predictions and all negative predictions.
- Matthew's correlation coefficient (MCC): this metric measures the correlation between true and predicted values. It may vary from -1 to 1 and the closer to 1 the correlation value is, the better is the predictions.
- F1 Score (F1): this metric is the harmonic mean of precision and recall. Precision indicates how many positive predictions are positive.

All the procedures were performed on a personal laptop with the following configurations: Linux, 64-bit Operating System, Intel Core processor i5-3210M 2.50GHz, and 8 GB of RAM. Moreover, R software version 3.6.3 with packages keras [56] and kernlab [57]. This same configuration was also used in the next section.

In the support vector models, the selection of the σ parameter of Gaussian kernel function from Table 1, was realized using an interval with lower and upper bound of the 0.1 and 0.9 quantiles of $||\boldsymbol{x} - \boldsymbol{x}'||^2$, respectively. Reference [58] showed that any values inside that interval will produce good values of σ. This procedure can exponentially reduce the computational time of Gaussian SVM, once exhaustive hyperparameter tuning is not needed to find acceptable σ values. For the other kernel functions presented in Table 1, the default parameters used were $q = 2$, $d = 0$, and $\gamma = 1$. Tables 3 and 4 summarize the simulation results of a cross validation 90/10 in 100 repetitions for each batch size. The values of each metric are the average of the repetitions.

We can observe that all models have poor performance for small batch size (less than 500) simulations, but CSVM models have smaller processing time than CNN. For batch size equal to 500 (Table 3), CSVM models and CNN$_1$ presented an improvement in the prediction capacity whilst CNN$_2$ model kept the same performance as in small size batches. Finally, in the larger image batch, all models have an expressive improvement in performance when compared to the results of small batches. For all image batch sizes, CSVM models present smaller processing times and competitive predictive performance. Regarding the results from Table 4, given that the difference between the means is equals to 3SD and the sample size is large enough, this is a simple task for machine learning algorithms to correctly classify the two classes, which is shown through the unusual results equal to one. However, even in those cases, the computational time of CSVM remains significantly smaller when compared with CNN.

Table 3. Simulation results for convolutional methods with μ difference equals to 1SD. The time column refers to the model's training time. The suitable model performance is highlighted in bold for each case.

Samples	Method	ACC	SEN	SPC	MCC	F1	Time
100	CSVM$_{Gau}$	0.46	0.40	0.51	−0.10	0.45	0.09
	CSVM$_{Lin}$	0.45	0.41	0.49	−0.10	0.44	0.09
	CSVM$_{Pol}$	0.45	0.41	0.49	−0.10	0.45	0.09
	CNN$_1$	**0.49**	**0.23**	**0.75**	**−0.04**	**0.50**	**0.12**
	CNN$_2$	0.49	0.11	0.88	−0.08	0.47	0.24
300	CSVM$_{Gau}$	0.52	0.50	0.54	0.05	0.50	0.10
	CSVM$_{Lin}$	0.54	0.54	0.54	0.08	0.53	0.10
	CSVM$_{Pol}$	**0.54**	**0.55**	**0.53**	**0.08**	**0.54**	**0.10**
	CNN$_1$	0.51	0.48	0.55	0.03	0.49	0.20
	CNN$_2$	0.50	0.08	0.91	−0.07	0.40	0.53
500	**CSVM**$_{Gau}$	**0.88**	**0.88**	**0.88**	**0.77**	**0.88**	**0.13**
	CSVM$_{Lin}$	0.88	0.87	0.88	0.76	0.88	0.12
	CSVM$_{Pol}$	0.87	0.86	0.88	0.74	0.87	0.12
	CNN$_1$	0.87	0.85	0.88	0.74	0.86	0.27
	CNN$_2$	0.50	0.01	0.99	0.29	0.70	0.87
1000	CSVM$_{Gau}$	0.98	0.98	0.98	0.97	0.98	0.24
	CSVM$_{Lin}$	0.98	0.98	0.99	0.97	0.98	0.20
	CSVM$_{Pol}$	0.98	0.98	0.98	0.97	0.98	0.22
	CNN$_1$	**0.99**	**0.99**	**0.99**	**0.98**	**0.99**	**0.50**
	CNN$_2$	0.77	0.55	0.98	0.93	0.97	2.46

Table 4. Simulation results for convolutional methods with μ difference equals to 3SD. The time column refers to the model's training time. The suitable model performance is highlighted in bold for each case.

Samples	Method	ACC	SEN	SPC	MCC	F1	Time
100	CSVM$_{Gau}$	0.45	0.41	0.49	−0.10	0.45	0.09
	CSVM$_{Lin}$	0.44	0.43	0.45	−0.13	0.45	0.10
	CSVM$_{Pol}$	0.44	0.42	0.46	−0.12	0.45	0.10
	CNN$_1$	0.48	0.22	0.74	−0.13	0.50	0.13
	CNN$_2$	**0.50**	**0.16**	**0.84**	**−0.12**	**0.56**	**0.24**
300	**CSVM**$_{Gau}$	**0.51**	**0.54**	**0.47**	**0.02**	**0.52**	**0.10**
	CSVM$_{Lin}$	0.51	0.51	0.50	0.19	0.51	0.10
	CSVM$_{Pol}$	0.51	0.52	0.51	0.03	0.51	0.10
	CNN$_1$	0.51	0.48	0.53	0.02	0.48	0.21
	CNN$_2$	0.49	0.17	0.82	−0.02	0.38	0.54

Table 4. *Cont.*

Samples	Method	ACC	SEN	SPC	MCC	F1	Time
500	CSVM_{Gau}	0.88	0.88	0.88	0.78	0.88	0.12
	CSVM_{Lin}	0.89	0.89	0.89	0.79	0.89	0.11
	$\textbf{CSVM}_{Pol}$	**0.90**	**0.89**	**0.90**	**0.80**	**0.90**	**0.11**
	CNN_1	0.89	0.89	0.89	0.79	0.89	0.28
	CNN_2	0.88	0.87	0.90	0.80	0.89	0.95
1000	CSVM_{Gau}	1.00	1.00	1.00	1.00	1.00	0.18
	$\textbf{CSVM}_{Lin}$	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**0.16**
	$\textbf{CSVM}_{Pol}$	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**0.16**
	CNN_1	1.00	1.00	1.00	1.00	1.00	0.48
	CNN_2	0.99	0.99	1.00	1.00	1.00	2.65

5. Real Data Study

5.1. Data and X-ray Image Acquisition

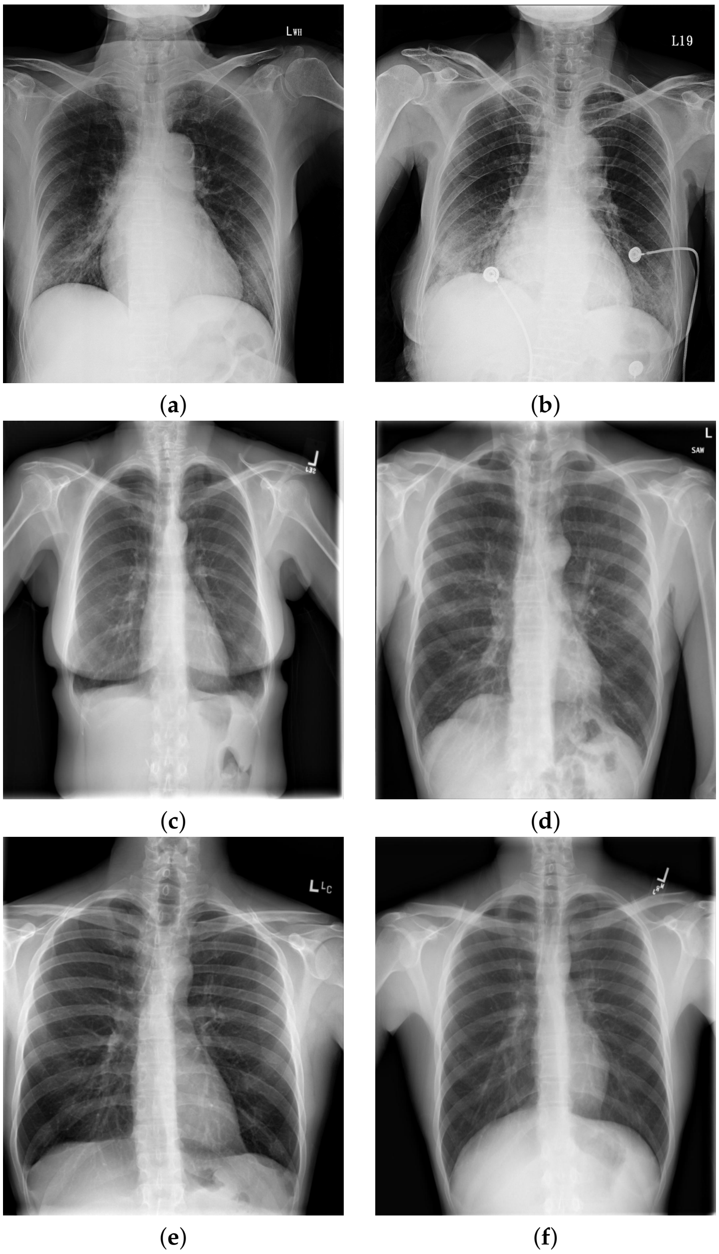
Images used in this analysis came from anteroposterior radiographs (X-rays). Radiography is a technique for generating and recording an X-ray pattern to provide the user with a static image(s) after the end of X-ray exposure, as displayed in Figure 12. The erect anteroposterior (AP) chest view is performed with an X-ray tube firing photons through the patient to form the image on a detector positioned behind the patient [59].

Image data were collected from two main sources: (i) COVID-19 and other lung disease cases came from COVID-19 Image Data Collection [60]; (ii) healthy X-ray images were retrieved from the Open Access Biomedical Search Engine (https://openi.nlm.nih.gov/). The COVID-19 Image Data Collection is the first public COVID-19 CXR data collection and aims to aid in the treatment of the disease. Data aggregation was partially made by hand, in the case of public research articles, or automated, in the case of images stored at websites (such as Radiopedia or Eurorad). From the set of images released on the GitHub (https://www.github.com/ieee8023/covid-chestxray-dataset), 105 images of patients diagnosed with COVID-19 were collected on 4 April 2020. A new search on 27 July 2020 has updated our dataset with 142 new images, including AP (anteroposterior) and AP Supine (patient laying down) images from different patients with confirmation of SARSr-CoV-2. After selecting the images, a hashing technique was used to detect duplicates and 30 duplicated images were removed, resulting in 217 COVID-19 detected X-rays. After removing duplicates, a grayscale/color conversion was applied to each of them through the OpenCV library in Python. This was the only pre-processing technique applied. No noise reduction methods were used. Another set of 108 images with other lung diseases (SARS, Pneumocystis, or Legionella) was gathered from the same GitHub database from COVID-19 Image Data Collection.

The query used for healthy chest X-ray images can be reproduced by accessing the Open-i service (https://openi.nlm.nih.gov). This service aims to enable the search and collection of abstracts and images from open-source literature. In our study, 105 images were retrieved on 4 April 2020, while seven new images from COVID-19 Image Data Collection with a “no Finding” tag were added, resulting in a dataset with 112 images. No duplicated images were found in this case. After the image retrieval, the final dataset consisted of 437 images as summarized in Table 5. Image sizes range from a minimal height and width of 235 and 256 pixels to a maximum of 4757 and 5623 pixels.

Table 5. Overview of the image dataset used in this work.

	COVID-19	Other Diseases	Healthy	Total
Quantity	217	108	112	437



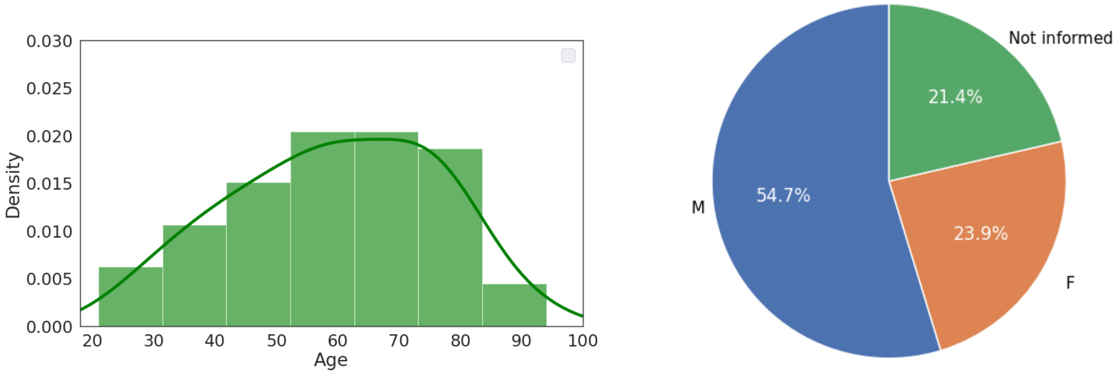
(**a**) (**b**)

(**c**) (**d**)

(**e**) (**f**)

Figure 12. Example of X-rays: (**a**,**b**) coronavirus (COVID-19); (**c**,**d**) other diseases; and (**e**,**f**) healthy. Source: prepared by the authors.

An exploratory analysis of the metadata provided by the COVID-19 Image Data Collection resulted in the data shown by Figure 13. Such metadata information was used in our predictive models since it is only available for COVID-19 cases. In addition, statistics metrics from patients' ages and sex information were calculated: mean $\bar{X}$ = 59.45, median $Med = 61$, and standard deviation σ = 16.88. Figure 12 displays an example of a healthy chest X-ray and an X-ray of a patient diagnosed with COVID-19. These images correspond to the dataset used to train our models. No data augmentation process was used.



(**a**) (**b**)

Figure 13. Metadata provided by the COVID-19 reference dataset. Age histogram (**a**) and gender pie chart (**b**). Source: prepared by the authors.

5.2. Predictive Models

The evaluation of all methods was realized through a repeated holdout validation technique, with a split ratio of the data of 90–10% in the training-test set and a total of 100 repetitions. Despite the most common validation technique used in image classification tasks being K-Fold, a high number of repetitions in the hold-out achieve lower bias and variance [61]. The classification performance of the CSVM models was compared with those from other methods in cluing traditional neural network (MLP_1 and MLP_2), in which all of them converged after 20 epochs. Table 6 presents the performance measured from ten methods ran 100 times each.

Table 6. Performance results. MLP1 corresponds to the model with more parameters, while MLP2 corresponds to the model with fewer parameters. The same applies to the CNN models. The time column refers to the training time.The suitable metrics are highlighted in bold.

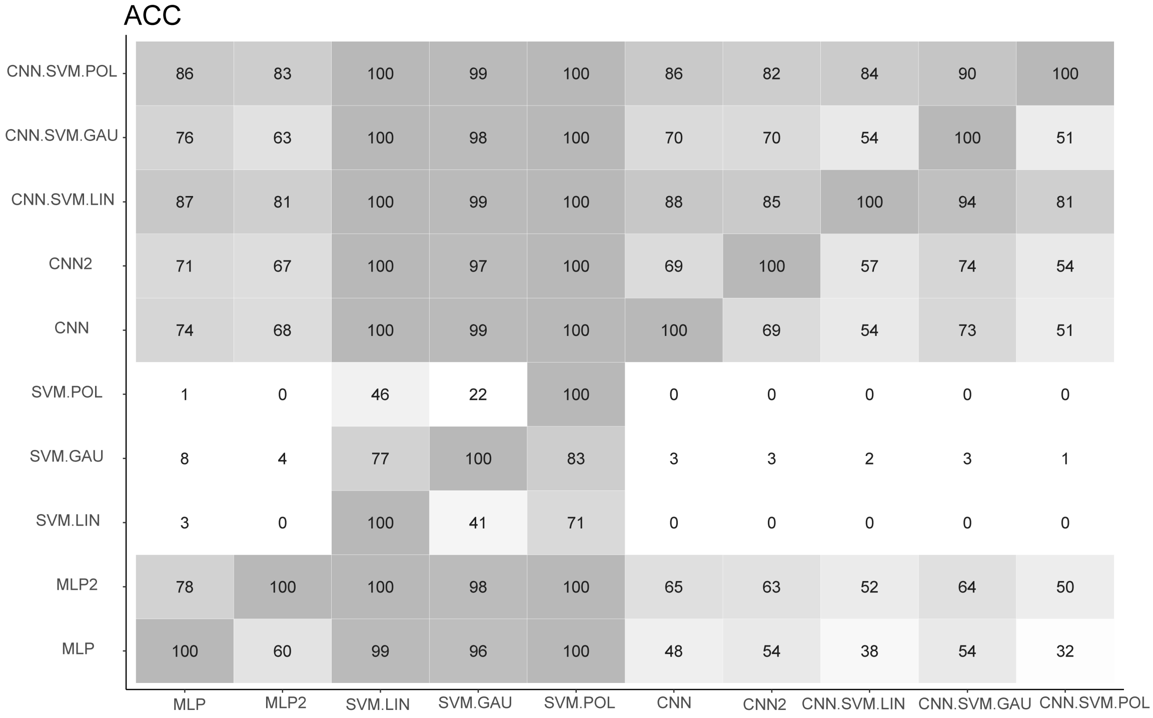
Method	ACC	F1	MCC	Time
MLP_1	95.54	95.46	91.57	0.0422
MLP_2	96.59	96.56	93.48	0.0370
CNN_1	96.67	96.63	93.48	0.7792
CNN_2	96.73	96.67	93.74	0.7585
SVM_{Lin}	80.79	80.21	61.98	0.0074
SVM_{Pol}	77.90	77.24	56.30	0.0076
SVM_{RBF}	83.45	83.86	67.39	**0.0067**
CSVM_{Lin}	98.00	97.97	96.11	0.0146
CSVM_{Pol}	96.57	**98.13**	**96.36**	0.0143
CSVM_{Gau}	**98.14**	96.59	93.34	0.0151

From Table 6, we can observe that the convolutional support vector machine with the polynomial kernel (CSVM_{Pol}) presented a better performance when compared with the others, achieving higher scores in F1 Score and MCC. The best accuracy was CSVM_{Gau} and SVM_{RBF} was the fastest among all models. Additionally, the CSVM_{Lin} achieved the second-best overall result, followed by MLP_2. From these outcomes, it can be observed in this situation that the support vector models are more useful from the convolution framework when compared with the traditional CNN.

In order to realize a complete comparison between the models, one important aspect is parsimony. The Occam's Razor [62], in the context of statistical modeling [63], states that "given two models with the same training set error, the simpler one should be preferred because it is likely to have lower generalization error". Regarding this statement, a model with a greater number of parameters is considered as being more complex. In general, the number of parameters of a SVM is given by the number of support vectors, and in the neural networks, these parameters correspond to the weights used in several nodes aggregation in the model. References [64–66] showed that the reduction of the number of support vectors would be able to produce parsimonious models yielding better results. The result of the SVM models produced respectively, on average, 201, 278, 264 while the from SVM$_{Lin}$, SVM$_{Pol}$, SVM$_{Gau}$ while the convolutional support vector models produced 31, 29, and 71 support vectors CSVM$_{Lin}$, CSVM$_{Pol}$, CSVM$_{Gau}$, respectively, showing that the convolution process was able to produce a more general model.

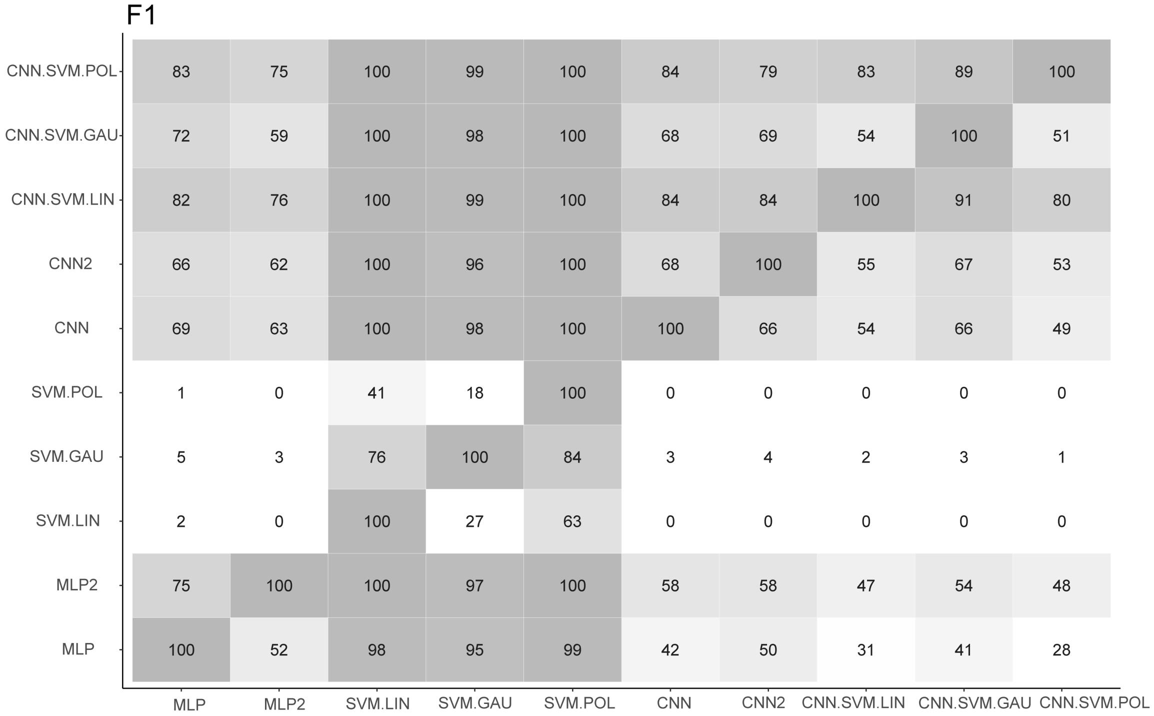
Another way to analyze and compare the performance of each algorithm is through a win-loss table, where it is counted the number of times that a method produced an equal or greater value of the determined evaluation metric. Figures 14 and 15 correspond, respectively, to accuracy (ACC) and F1 Score comparison. From all figures, we observed that those results reflect the previous table, where we can see that $CSVM_{Pol}$ produced the highest values in most of the times.

The computational effort from CNN and CSVM is an important aspect to be treated. Reference [67] presented that it is computationally expensive to train the convolutional neural networks, being necessary the use of GPU processing to train some models in a viable way.



	MLP	MLP2	SVM.LIN	SVM.GAU	SVM.POL	CNN	CNN2	CNN.SVM.LIN	CNN.SVM.GAU	CNN.SVM.POL
CNN.SVM.POL	86	83	100	99	100	86	82	84	90	100
CNN.SVM.GAU	76	63	100	98	100	70	70	54	100	51
CNN.SVM.LIN	87	81	100	99	100	88	85	100	94	81
CNN2	71	67	100	97	100	69	100	57	74	54
CNN	74	68	100	99	100	100	69	54	73	51
SVM.POL	1	0	46	22	100	0	0	0	0	0
SVM.GAU	8	4	77	100	83	3	3	2	3	1
SVM.LIN	3	0	100	41	71	0	0	0	0	0
MLP2	78	100	100	98	100	65	63	52	64	50
MLP	100	60	99	96	100	48	54	38	54	32

Figure 14. Number of times which a method obtained greater accuracy (ACC) than others. The count summarizes 100 holdout values. It is clear the superiority of CSVM$_{Pol}$ when it is compared with the other models. The darker is the table cell, the bigger is its score. Source: prepared by the authors.



	MLP	MLP2	SVM.LIN	SVM.GAU	SVM.POL	CNN	CNN2	CNN.SVM.LIN	CNN.SVM.GAU	CNN.SVM.POL
CNN.SVM.POL	83	75	100	99	100	84	79	83	89	100
CNN.SVM.GAU	72	59	100	98	100	68	69	54	100	51
CNN.SVM.LIN	82	76	100	99	100	84	84	100	91	80
CNN2	66	62	100	96	100	68	100	55	67	53
CNN	69	63	100	98	100	100	66	54	66	49
SVM.POL	1	0	41	18	100	0	0	0	0	0
SVM.GAU	5	3	76	100	84	3	4	2	3	1
SVM.LIN	2	0	100	27	63	0	0	0	0	0
MLP2	75	100	100	97	100	58	58	47	54	48
MLP	100	52	98	95	99	42	50	31	41	28

Figure 15. Number of times in which a method obtained greater F1 Score than the others. The count summarizes 100 holdout values. It is clear the superiority of CSVM$_{F1}$ when it is compared with the other models. The darker is the table cell, the bigger is its score. Source: prepared by the authors.

6. Final Considerations

This paper presented a novel study and application of the convolutional support vector machines to classify patients infected with COVID-19 using X-ray data. The result was compared with the use of a CNN approach for this task [36,38,39], which is considered the state-of-art in many image classification tasks [68–71]. The result showed that the CSVM outperformed the CNN approach through higher values of ACC, F1 Score, and MCC obtained in a holdout repetition as a robust validation procedure. The one-hundred repeated hold-out technique was used to reinforce that great performance as [61] showed that this validation procedure has lower bias and variance when compared with the commonly used K-Fold. In addition, the CSVM showed up as being computationally cheaper since it can be one-hundred times faster, on average, when compared with CNN in this situation with a medium sample size. In comparison with other works that presented analysis on detecting SARS-CoV-2 from X-ray images, we presented a novel database with the most recent data that is composed of more instances, and with a greater diversity of diseases in the group of no-COVID, improving the capacity of the model to generalize and predict new observations. Additionally, even by boosting research on COVID-19, data on such patients are still small or medium-scale, a fact that makes CSVM even more effective for these cases.

For future works, the number of kernel functions, as well as its hyperparameters can be explored to achieve even stronger results in CSVM. In addition, the architectures in both cases, CSVM and CNN, can be diversified.

Author Contributions: Conceptualization, A.A. and M.E.B.; data curation, M.M., J.S.P. and J.G.; formal analysis, M.M., I.S.P.; investigation, J.S.P. and I.S.P.; methodology, A.A.; project administration, A.A.; software, M.M., J.S.P., I.S.P., J.G. and A.A.; supervision, M.E.B. and A.A.; writing—original draft, M.M., J.S.P., I.S.P., M.E.B. and A.A.; writing—review and editing, M.E.B. and A.A. All authors have read and agreed to the published version of the manuscript.

Funding: M.M.'s work was supported by a Science Foundation Ireland Career Development Award grant 17/CDA/4695. The authors are grateful for the partial funding provided by the Brazilian agencies CNPq and CAPES.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, C.; Horby, P.W.; Hayden, F.G.; Gao, G.F. A novel coronavirus outbreak of global health concern. *Lancet* **2020**, *395*, 470–473. [CrossRef]
2. Dong, E.; Du, H.; Gardner, L. An interactive web-based dashboard to track COVID-19 in real time. *Lancet Infect. Dis.* **2020**, *20*, 533–534. [CrossRef]
3. Korber, B.; Fischer, W.M.; Gnanakaran, S.; Yoon, H.; Theiler, J.; Abfalterer, W.; Hengartner, N.; Giorgi, E.E.; Bhattacharya, T.; Foley, B.; et al. Tracking changes in SARS-CoV-2 Spike: Evidence that D614G increases infectivity of the COVID-19 virus. *Cell* **2020**, *182*, 812–827. [CrossRef] [PubMed]
4. Velavan, T.P.; Meyer, C.G. The COVID-19 epidemic. *Trop. Med. Int. Health* **2020**, *25*, 278. [CrossRef] [PubMed]
5. Sun, P.; Lu, X.; Xu, C.; Sun, W.; Pan, B. Understanding of COVID-19 based on current evidence. *J. Med. Virol.* **2020**, *92*, 548–551. [CrossRef] [PubMed]
6. Kim, J.Y.; Choe, P.G.; Oh, Y.; Oh, K.J.; Kim, J.; Park, S.J.; Park, J.H.; Na, H.K.; Oh, M.D. The first case of 2019 novel coronavirus pneumonia imported into Korea from Wuhan, China: Implication for infection prevention and control measures. *J. Korean Med. Sci.* **2020**, *35*, e61. [CrossRef]
7. Zhang, G.; Wang, W.; Moon, J.; Pack, J.K.; Jeon, S.I. A review of breast tissue classification in mammograms. In Proceedings of the 2011 ACM Symposium on Research in Applied Computation, Taichung, Taiwan, 21–25 March 2011; pp. 232–237.
8. El-Yaagoubi, M.; Mora-Jiménez, I.; Jabrane, Y.; Muñoz-Romero, S.; Rojo-Álvarez, J.L.; Pareja-Grande, J.A. Quantitative Cluster Headache Analysis for Neurological Diagnosis Support Using Statistical Classification. *Information* **2020**, *11*, 393. [CrossRef]
9. Pellegrini, E.; Ballerini, L.; Hernandez, M.D.C.V.; Chappell, F.M.; González-Castro, V.; Anblagan, D.; Danso, S.; Muñoz-Maniega, S.; Job, D.; Pernet, C.; et al. Machine learning of neuroimaging for assisted diagnosis of cognitive impairment and dementia: A systematic review. *Alzheimer's Dement. Diagn. Assess. Dis. Monit.* **2018**, *10*, 519–535. [CrossRef]
10. Yassin, N.I.; Omran, S.; El Houby, E.M.; Allam, H. Machine learning techniques for breast cancer computer aided diagnosis using different image modalities: A systematic review. *Comput. Methods Programs Biomed.* **2018**, *156*, 25–45. [CrossRef]
11. Asri, H.; Mousannif, H.; Al Moatassime, H.; Noel, T. Using machine learning algorithms for breast cancer risk prediction and diagnosis. *Procedia Comput. Sci.* **2016**, *83*, 1064–1069. [CrossRef]
12. Safdar, S.; Zafar, S.; Zafar, N.; Khan, N.F. Machine learning based decision support systems (DSS) for heart disease diagnosis: A review. *Artif. Intell. Rev.* **2018**, *50*, 597–623. [CrossRef]
13. Liu, S.; Wang, Y.; Yang, X.; Lei, B.; Liu, L.; Li, S.X.; Ni, D.; Wang, T. Deep learning in medical ultrasound analysis: A review. *Engineering* **2019**, *5*, 261–275. [CrossRef]
14. Bakator, M.; Radosav, D. Deep learning and medical diagnosis: A review of literature. *Multimodal Technol. Interact.* **2018**, *2*, 47. [CrossRef]
15. Shen, D.; Wu, G.; Suk, H.I. Deep learning in medical image analysis. *Annu. Rev. Biomed. Eng.* **2017**, *19*, 221–248. [CrossRef] [PubMed]
16. Cortes, C.; Vapnik, V. Support Vector Networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
17. Tzotsos, A.; Argialas, D. Support vector machine classification for object-based image analysis. In *Object-Based Image Analysis*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 663–677.
18. Song, Q.; Hu, W.; Xie, W. Robust support vector machine with bullet hole image classification. *IEEE Trans. Syst. Man. Cybern. Part C Appl. Rev.* **2002**, *32*, 440–448. [CrossRef]
19. Chaplot, S.; Patnaik, L.M.; Jagannathan, N. Classification of magnetic resonance brain images using wavelets as input to support vector machine and neural network. *Biomed. Signal Process. Control* **2006**, *1*, 86–92. [CrossRef]
20. Maulik, U.; Chakraborty, D. Remote Sensing Image Classification: A survey of support-vector-machine-based advanced techniques. *IEEE Geosci. Remote Sens. Mag.* **2017**, *5*, 33–52. [CrossRef]
21. Islam, M.; Dinh, A.; Wahid, K.; Bhowmik, P. Detection of potato diseases using image segmentation and multiclass support vector machine. In Proceedings of the 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), Windsor, ON, Canada, 30 April–3 May 2017; pp. 1–4.

22. Huang, F.J.; LeCun, Y. Large-scale Learning with SVM and Convolutional Nets for Generic Object Categorization. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–22 June 2006; pp. 1–8.
23. Şentaş, A.; Tashiev, İ.; Küçükayvaz, F.; Kul, S.; Eken, S.; Sayar, A.; Becerikli, Y. Performance evaluation of support vector machine and convolutional neural network algorithms in real-time vehicle type and color classification. *Evol. Intell.* **2020**, *13*, 83–91. [CrossRef]
24. Chagas, P.; Souza, L.; Araújo, I.; Aldeman, N.; Duarte, A.; Angelo, M.; dos Santos, W.L.; Oliveira, L. Classification of glomerular hypercellularity using convolutional features and support vector machine. *Artif. Intell. Med.* **2020**, *103*, 101808. [CrossRef]
25. Witoonchart, P.; Chongstitvatana, P. Application of structured support vector machine backpropagation to a convolutional neural network for human pose estimation. *Neural Netw.* **2017**, *92*, 39–46. [CrossRef] [PubMed]
26. Zafar, R.; Malik, A.S.; Shuaibu, A.N.; ur Rehman, M.J.; Dass, S.C. Classification of fmri data using support vector machine and convolutional neural network. In Proceedings of the 2017 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), Kuching, Malaysia, 12–14 September 2017; pp. 324–329.
27. Li, M.; Han, C.; Fahim, F. Skin Cancer Diagnosis Based on Support Vector Machine and a New Optimization Algorithm. *J. Med. Imaging Health Inform.* **2020**, *10*, 356–363. [CrossRef]
28. Ferreira, L.K.; Rondina, J.M.; Kubo, R.; Ono, C.R.; Leite, C.C.; Smid, J.; Bottino, C.; Nitrini, R.; Busatto, G.F.; Duran, F.L.; et al. Support vector machine-based classification of neuroimages in Alzheimer's disease: Direct comparison of FDG-PET, rCBF-SPECT and MRI data acquired from the same individuals. *Braz. J. Psychiatry* **2018**, *40*, 181–191. [CrossRef] [PubMed]
29. Kale, S.D.; Punwatkar, K.M. Texture analysis of ultrasound medical images for diagnosis of thyroid nodule using support vector machine. *Int. J. Comput. Sci. Mob. Comput.* **2013**, *2*, 71–77.
30. Orru, G.; Pettersson-Yeo, W.; Marquand, A.F.; Sartori, G.; Mechelli, A. Using support vector machine to identify imaging biomarkers of neurological and psychiatric disease: A critical review. *Neurosci. Biobehav. Rev.* **2012**, *36*, 1140–1152. [CrossRef]
31. Novitasari, D.C.R.; Hendradi, R.; Caraka, R.E.; Rachmawati, Y.; Fanani, N.Z.; Syarifudin, A.; Toharudin, T.; Chen, R.C. Detection of COVID-19 chest X-ray using support vector machine and convolutional neural network. *Commun. Math. Biol. Neurosci.* **2020**, *2020*. [CrossRef]
32. Sethy, P.K.; Behera, S.K. Detection of coronavirus disease (covid-19) based on deep features. *Preprints* **2020**, 030300. [CrossRef]
33. Tayarani-N, M.H. Applications of Artificial Intelligence in Battling Against Covid-19: A Literature Review. *Chaos Solitons Fractals* **2020**, 110338. [CrossRef]
34. Chen, D.; Liu, F.; Li, Z. A Review of Automatically Diagnosing COVID-19 based on Scanning Image. *arXiv* **2020**, arXiv:2006.05245
35. Nishio, M.; Noguchi, S.; Matsuo, H.; Murakami, T. Automatic classification between COVID-19 pneumonia, non-COVID-19 pneumonia, and the healthy on chest X-ray image: Combination of data augmentation methods in a small dataset. *arXiv* **2020**, arXiv:2006.00730.
36. Luz, E.J.D.S.; Silva, P.L.; Silva, R.; Silva, L.; Moreira, G.; Menotti, D. Towards an Effective and Efficient Deep Learning Model for COVID-19 Patterns Detection in X-ray Images. *arXiv* **2020**, arXiv:2004.05717.
37. Apostolopoulos, I.D.; Mpesiana, T.A. Covid-19: Automatic detection from x-ray images utilizing transfer learning with convolutional neural networks. *Phys. Eng. Sci. Med.* **2020**, *43*, 635–640. [CrossRef] [PubMed]
38. Heidari, M.; Mirniaharikandehei, S.; Khuzani, A.Z.; Danala, G.; Qiu, Y.; Zheng, B. Improving the performance of CNN to predict the likelihood of COVID-19 using chest X-ray images with preprocessing algorithms. *Int. J. Med. Inform.* **2020**, *144*, 104284. [CrossRef] [PubMed]
39. Cao, K.; Choi, K.N.; Jung, H.; Duan, L. Deep Learning for Facial Beauty Prediction. *Information* **2020**, *11*, 391. [CrossRef]
40. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA , USA, 2016.
41. Witten, I.H.; Frank, E.; Hall, M.A. *Data Mining Practical Learning Tools and Techniques*, 4rd ed.; Morgan Kaufmann: Burlington, MA, USA, 2017; pp. 417–466.

42. Wang, B.; Sun, Y.; Xue, B.; Zhang, M. Evolving Deep Convolutional Neural Networks by Variable-length Particle Swarm Optimization for Image Classification. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018.
43. Ciaburro, G.; Venkateswaran, B. *Neural Networks with R*; Packt Publishing: Birmingham, UK, 2017.
44. Haykin, S. *Neural Networks and Learning Machines*, 3rd ed.; Pearson Education, Inc.: Upper Saddle River, NJ, USA, 2009.
45. Nagi, J.; Di Caro, G.A.; Giusti, A.; Nagi, F.; Gambardella, L.M. Convolutional Neural Support Vector Machines: Hybrid visual pattern classifiers for multirobot systems. In Proceedings of the 2012 11th International Conference on Machine Learning and Applications, Boca Raton, FL, USA, 12–15 December 2012.
46. Tang, Y. Deep Learning using Linear Support Vector Machines. *arXiv* **2015**, arXiv:1306.0239.
47. Vapnik, V.N. An overview of statistical learning theory. *IEEE Trans. Neural Netw.* **1999**, *10*, 988–999. [CrossRef]
48. Wang, W.; Xu, Z.; Lu, W.; Zhang, X. Determination of the spread parameter in the Gaussian kernel for classification and regression. *Neurocomputing* **2003**, *55*, 643–663. [CrossRef]
49. Yaohao, P. Support Vector Regression Aplicado à Previsão de Taxas de Câmbio. Master's Thesis, Universidade de Brasilia, Brasilia, Brazil,2016.
50. Elangovan, M.; Sugumaran, V.; Ramachandran, K.; Ravikumar, S. Effect of SVM kernel functions on classification of vibration signals of a single point cutting tool. *Expert Syst. Appl.* **2011**, *38*, 15202–15207. [CrossRef]
51. Yekkehkhany, B.; Safari, A.; Homayouni, S.; Hasanlou, M. A comparison study of different kernel functions for SVM-based classification of multi-temporal polarimetry SAR data. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2014**, *40*, 281. [CrossRef]
52. Chen, H.; Chen, A.; Xu, L.; Xie, H.; Qiao, H.; Lin, Q.; Cai, K. A deep learning CNN architecture applied in smart near-infrared analysis of water pollution for agricultural irrigation resources. *Agric. Water Manag.* **2020**, *240*, 106303. [CrossRef]
53. Sarmento, P.L. Avaliação de méTodos de Seleção de Amostras para Redução do Tempo de Treinamento do Classificador SVM. Master's Thesis, INPE, Sao Jose dos Campos, Brazil, 2014.
54. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the NIPS 2012, Lake Tahoe, NV, USA, 3–6 December 2012; Volume 25, pp. 1097–1105.
55. Baldi, P.; Brunak, S.; Chauvin, Y.; Andersen, C.A.; Nielsen, H. Assessing the accuracy of prediction algorithms for classification: An overview. *Bioinformatics* **2000**, *16*, 412–424. [CrossRef] [PubMed]
56. Chollet, F.; Allaire, J. R Interface to Keras. 2017. Available online: https://github.com/rstudio/keras (accessed on 20 November 2020).
57. Karatzoglou, A.; Smola, A.; Hornik, K.; Zeileis, A. kernlab—An S4 Package for Kernel Methods in R. *J. Stat. Softw.* **2004**, *11*, 1–20. [CrossRef]
58. Caputo, B.; Sim, K.; Furesjo, F.; Smola, A. Appearance-based object recognition using SVMs: Which kernel should I use? In Proceedings of the NIPS Workshop on Statistical Methods for Computational Experiments in Visual Processing and Computer Vision, Vancouver, BC, Canada, 9–14 December 2002; Volume 2002.
59. Radiopedia. Chest (AP Erect View). Available online: https://radiopaedia.org/articles/chest-ap-erect-view-1 (accessed on 15 August 2020).
60. Cohen, J.P.; Morrison, P.; Dao, L.; Roth, K.; Duong, T.Q.; Ghassemi, M. COVID-19 Image Data Collection: Prospective Predictions Are the Future. *arXiv* **2020**, arXiv:2006.11988.
61. Tantithamthavorn, C.; McIntosh, S.; Hassan, A.E.; Matsumoto, K. An empirical comparison of model validation techniques for defect prediction models. *IEEE Trans. Softw. Eng.* **2016**, *43*, 1–18. [CrossRef]
62. Blumer, A.; Ehrenfeucht, A.; Haussler, D.; Warmuth, M.K. Occam's razor. *Inf. Process. Lett.* **1987**, *24*, 377–380. [CrossRef]
63. Domingos, P. The role of Occam's razor in knowledge discovery. *Data Min. Knowl. Discov.* **1999**, *3*, 409–425. [CrossRef]
64. Osuna, E.; Freund, R.; Girosi, F. An improved training algorithm for support vector machines. In *Neural Networks for Signal Processing VII, Proceedings of the 1997 IEEE Signal Processing Society Workshop, Amelia Island, FL, USA, USA, 24–26 September 1997*; IEEE: Piscataway, NJ, USA, 1997; pp. 276–285.
65. Downs, T.; Gates, K.E.; Masters, A. Exact simplification of support vector solutions. *J. Mach. Learn. Res.* **2001**, *2*, 293–297.

66. Geebelen, D.; Suykens, J.A.; Vandewalle, J. Reducing the number of support vectors of SVM classifiers using the smoothed separable case approximation. *IEEE Trans. Neural Netw. Learn. Syst.* **2012**, *23*, 682–688. [CrossRef]
67. Kim, H.; Nam, H.; Jung, W.; Lee, J. Performance analysis of CNN frameworks for GPUs. In Proceedings of the 2017 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Santa Rosa, CA, USA, 24–25 April 2017; pp. 55–64.
68. Litjens, G.; Ciompi, F.; Wolterink, J.M.; de Vos, B.D.; Leiner, T.; Teuwen, J.; Išgum, I. State-of-the-art deep learning in cardiovascular image analysis. *JACC Cardiovasc. Imaging* **2019**, *12*, 1549–1565. [CrossRef]
69. Pound, M.P.; Atkinson, J.A.; Townsend, A.J.; Wilson, M.H.; Griffiths, M.; Jackson, A.S.; Bulat, A.; Tzimiropoulos, G.; Wells, D.M.; Murchie, E.H.; et al. Deep machine learning provides state-of-the-art performance in image-based plant phenotyping. *Gigascience* **2017**, *6*, gix083. [CrossRef]
70. Mazurowski, M.A.; Buda, M.; Saha, A.; Bashir, M.R. Deep learning in radiology: An overview of the concepts and a survey of the state of the art with focus on MRI. *J. Magn. Reson. Imaging* **2019**, *49*, 939–954. [CrossRef] [PubMed]
71. Cheng, G.; Han, J.; Lu, X. Remote sensing image scene classification: Benchmark and state of the art. *Proc. IEEE* **2017**, *105*, 1865–1883. [CrossRef]

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.