Research on Biomedical Engineering (2022) 38:139–148
https://doi.org/10.1007/s42600-021-00132-9

ORIGINAL ARTICLE

A deep convolutional neural network for COVID-19 detection using chest X-rays

Pedro R. A. S. Bassi[1] · Romis Attux[1]

Received: 22 June 2020 / Accepted: 8 March 2021 / Published online: 2 April 2021

Abstract

Purpose We present image classifiers based on Dense Convolutional Networks and transfer learning to classify chest X-ray images according to three labels: COVID-19, pneumonia, and normal.

Methods We fine-tuned neural networks pretrained on ImageNet and applied a twice transfer learning approach, using NIH ChestX-ray14 dataset as an intermediate step. We also suggested a novelty called output neuron keeping, which changes the twice transfer learning technique. In order to clarify the modus operandi of the models, we used Layer-wise Relevance Propagation (LRP) to generate heatmaps.

Results We were able to reach test accuracy of 100% on our test dataset. Twice transfer learning and output neuron keeping showed promising results improving performances, mainly in the beginning of the training process. Although LRP revealed that words on the X-rays can influence the networks' predictions, we discovered this had only a very small effect on accuracy.

Conclusion Although clinical studies and larger datasets are still needed to further ensure good generalization, the state-of-the-art performances we achieved show that, with the help of artificial intelligence, chest X-rays can become a cheap and accurate auxiliary method for COVID-19 diagnosis. Heatmaps generated by LRP improve the interpretability of the deep neural networks and indicate an analytical path for future research on diagnosis. Twice transfer learning with output neuron keeping improved DNN performance.

Keywords COVID-19 detection · Neural networks · Chest X-ray · LRP · Twice transfer learning · Output neuron keeping

Introduction

In 2020, COVID-19 became pandemic, affecting both developed and developing countries around the world. By October 2020, the virus had already infected more than 40,000,000 people and caused more than one million deaths (Hopkins (2020)).

The most commonly used method for COVID-19 diagnosis is reverse transcriptase-polymerase chain reaction (RT-PCR) (Wang et al. (2020)). It has a high specificity, but is also expensive, slow and currently at a high demand. Chest X-rays are commonly available and are faster and cheaper, but signals associated with the presence of COVID-19 in the lungs can be hard to detect.

Researchers have already suggested the use of deep neural networks (DNNs) to help in the detection of the disease on Chest X-ray images (Wang and Wong (2020), Shoeibi et al. (2020)). In Wang and Wong (2020), the authors achieved good results, with 92.6% test accuracy, 96.4% recall and 87% precision on the COVID-19 images.

Deep neural networks (DNNs) have been successful at identifying pneumonia from X-rays, performing better than radiologists (Rajpurkar et al. (2017)). In this work, we used Dense Convolutional Networks or DenseNets (Huang et al. (2016)). The first network is CheXNet (Rajpurkar et al. (2017)), a 121 layers dense network (or DenseNet121) that had already been pretrained on ImageNet (Deng et al. (2009)) and on NIH ChestX-ray14 dataset (Wang et al. (2017)), a database with over 100,000 frontal X-ray images, which contain 14 different diseases and also healthy individuals. We applied transfer learning to teach the neural network

✉ Pedro R. A. S. Bassi
p157007@dac.unicamp.br

Romis Attux
attux@dca.fee.unicamp.br

[1] Department of Computer Engineering and Industrial Automation, School of Electrical and Computer Engineering, University of Campinas, Campinas, Brazil

to differentiate between normal lungs, COVID-19 and pneumonia. Our COVID-19 dataset was assembled by merging COVID-19, pneumonia and normal chest X-ray open datasets. Other DNN is a 201 layers DenseNet that had been pretrained on ImageNet (Deng et al. (2009)) and we also fine tuned it in our COVID-19 database.

CheXNet is a neural network that had been trained twice (on ImageNet and ChestX-ray14) and we trained it again, making our process a twice transfer learning or transfer learning in three steps. Inspired by this, we decided to explore the technique. We downloaded a 201-layer DenseNet, already pretrained on ImageNet, trained it on NIH ChestX-ray14 dataset (Wang et al. (2017)) and then on our smaller dataset containing the COVID-19 class.

In this paper, we also suggest an original modification to twice transfer learning, which we called "output neuron keeping." As the NIH ChestX-ray14 database already had the healthy and pneumonia classes, we suggest keeping the DNN output neurons for these classes in the last step of twice transfer learning (training on the COVID-19 dataset). Our hypothesis is that this will enable us to keep more of the information learned in the second dataset (ChestX-ray14) throughout training in the COVID-19 database, and in the final network. We tested this approach with another DenseNet201 and a CheXNet (maintaining only the neuron that classified pneumonia in this last case).

After training the DNNs, we applied Layer-wise Relevance Propagation (LRP) (Bach et al. (2015)), generating heatmaps of the X-rays, along with the probabilities of COVID-19, pneumonia and healthy lungs. These heatmaps show us the regions of the image that mostly influenced the network classification, and also regions that were more representative of other classes. Therefore, LRP can allow us to understand what the DNN finds relevant in an input image. But it can also be useful for a radiologist in identifying the effects of COVID-19 in the X-ray. An application of LRP in the context of neuroimaging can be seen at Thomas et al. (2019).

Methods

Databases

NIH ChestX-ray14

ChestX-ray14 is one of the largest chest X-ray datasets, with 112,120 images from 30,805 patients. The images were obtained by the US National Institutes of Health. It has 14 different diseases and also images with no findings. Some X-rays may show multiple conditions, making the classification of this dataset a multi-label classification problem. The database images originally had associated radiological reports, which were analyzed by the dataset authors, using natural language processing, to create automated labels for the X-rays. These labels, which will be used in this work, have an estimated accuracy higher than 90% (Wang et al. (2017)). The dataset is unbalanced. State-of-the-art pneumonia-detecting DNNs were trained in this database: as an example, we can cite CheXNet (Rajpurkar et al. (2017)), a DenseNet with 121 layers.

COVID-19 database

This database was assembled by the research group, joining images from 3 chest X-ray datasets, the first containing COVID-19 images, the second containing pneumonia images, and the third with healthy lungs images.

We obtained 439 COVID-19 frontal X-rays from the dataset "Covid-19 image data collection" (Cohen et al. (2020)), downloaded in October 2020. To select these images we started with all COVID-19 frontal chest X-rays from Cohen et al. (2020) and excluded images with arrows. This dataset was the largest COVID-19 X-ray collection that we could find. It is also one of the best documented datasets, many images contained information about the patient age, gender, disease severity and image source. Their X-rays were collected from public sources or indirectly from hospitals and physicians (Cohen et al. (2020)).

The pneumonia images were extracted from the CheXpert database. It is a collection of 224,316 chest X-rays, from 65,240 patients, classified according to 13 lung diseases or healthy (Irvin et al. (2019)). We obtained 1255 images classified as pneumonia and whose patients were older than 18 years old. Because the reported youngest COVID-19 patient in our dataset was 20 years old, we decided that keeping children in the other classes could be a source of bias. Like in NIH ChestX-ray14, the images in this dataset were automatically labeled by the database authors, using natural language processing to analyze radiological reports. These labels accuracy are also estimated to be higher than 90%. The exceptions are 8 pneumonia images, which were manually labeled by three board-certified radiologists. These images were on the original CheXpert test dataset. The X-rays were obtained in the Stanford University Hospital.

The healthy images were obtained from the Montgomery and Shenzen databases (Jaeger et al. (2014)). They are a collection of tuberculosis and healthy frontal chest X-rays, labeled by radiologists. The images were obtained from the Department of Health and Human Services, Montgomery County (Maryland, USA) and Shenzhen No. 3 People's Hospital in China (Jaeger et al. (2014)). To select our images, we separated all normal X-rays and removed the ones whose patients were underage. Therefore, we ended up with 370 normal images.

The labels that we used in our COVID-19 database had been created by the authors of the above mentioned datasets

and were provided along with the X-rays. Therefore, the "healthy" and "COVID-19" labels were created in accordance with radiological reports that accompanied the images, while most of the pneumonia labels were produced by using natural language processing to analyze radiological reports. The exceptions are 8 pneumonia labels that were manually created by three radiologists (we included these images in our test dataset, which will be explained in the section "Data processing and Augmentation").

Additional details of the assembled COVID-19 database

In this section, we will describe some general information about the patients in our COVID-19 database.

There were 370 healthy patients (the database from Jaeger et al. (2014) does not contain multiple images from the same patient, according to the authors). The patients are 61.9% male and their mean age is 36.1 years, with a standard deviation of 12.3 years.

Our pneumonia images were created with 1047 different patients. They are 58.2% male, with a mean age of 61.8 years and an age standard deviation of 19.3 years.

Our COVID-19 database contained 268 patients, 246 had gender information and 199 had age information. Their mean age was 42.8 years, with standard deviation of 16.4 years. They were 64.2% male.

We also have information about disease severity in some COVID-19 patients. We have survival information about 81 patients, with a survival rate of 81.5%. We have information about intubation on 69 patients, with an intubation rate of 59.4%. We have information about supplemental oxygen on 94 patients, with 59.6% of them needing it. At last, we have ICU information on 112 patients, 59.8% of them were in the intensive care unit.

Transfer learning, twice transfer learning, and output neuron keeping

Introduction to transfer learning

When we add dimensions to a neural network input, the data tends to become sparser. Thus, with larger inputs (like our 224×224 images), we need more data to create a good statistical model of the inputs and labels distribution. This problem is known as the "curse of dimensionality" (Trunk (1979)).

Deep neural networks are mathematical models with many trainable parameters, enabling them to model complex data distributions and statistical relations. But, when we do not have enough data, this also makes them prone to learn small variations and noise in the training dataset, which are exclusive to that database and do not reflect the real phenomenon we are trying to model. Thus, with insufficient data, we can generate overfitting, hence creating a neural network that performs well on the training dataset but badly on the test database (Goodfellow et al. (2016)).

In summary, DNNs have a tendency to overfit when trained on small datasets and large inputs. Transfer learning is a technique that helps to avoid this problem. It consists in using a network that was already trained to solve a task in one dataset, and training it again (or fine-tuning) on another database, to solve another task. Doing this, we hope that representations learned by the DNN in the first database can help the model generalization on the second. This is particularly helpful when the first dataset is much larger than the second one (Goodfellow et al. (2016)).

When we train a deep neural network, each layer learns to map the information it receives onto a new representation of the input data, creating what is called representation learning (Bengio et al. (2012)). Thus, the layers implicitly extract features of the inputs. The nearer from the network output a layer is, the higher the level of abstraction the learned feature has (Goodfellow et al. (2016)). What makes transfer learning effective is that some features, learned from the first task and dataset, can help the DNN solve the second task, in the second dataset. For example, a network trained with a large image classification dataset, like ImageNet (Deng et al. (2009)), can learn, in that database, to identify image edges, a feature that can also be useful for interpreting X-rays, in a dataset like ChestXray-14.

We can observe that, if the two tasks are similar, more features learned in the first dataset will be useful in the second one, increasing the benefit of transfer learning. Thus, an ideal case would be to have a first dataset that is very large and whose task is very similar to that of the second one.

When we use an already trained network in another dataset, we need to pay attention to whether the input size remains the same (if not, the inputs are generally re-scaled). Also, because we are changing the task, the DNN output needs to change. One can add new layers at the end of the network or replace the last layers with new ones (Goodfellow et al. (2016)). A common approach is to replace just the output layer, removing it and adding a new one, with randomized weights and biases. The more similar the two tasks are, the more the last layers learned representations will be useful in the second dataset, and the more we would want to keep them.

Twice transfer learning

It is common to choose ImageNet as the first dataset when we have image classification tasks (like in Rajpurkar et al. (2017)), as it is a database with millions of images and many classes. But one can argue that classifying these images is not a task particularly similar to that of classifying chest X-rays as COVID-19, pneumonia or normal.

NIH ChestX-ray14 classification was a task much more alike ours, and the dataset is still very large, with over

100,000 X-rays. Also, beginning with a DNN already pretrained on ImageNet would accelerate training on the NIH database and the network could keep some information, learned in ImageNet, through training on ChestX-ray14. This information might also help in the final fine-tuning, on the COVID-19 dataset.

Thus, a twice transfer learning, or transfer learning in three steps seemed like a good proposition: a DNN would be first trained on ImageNet (Denget al. (2009)), then on ChestX-ray14 (Wang et al. (2017)) and, finally, on the COVID-19 dataset that we assembled. Fine-tuning CheXNet in the COVID-19 dataset indirectly created a transfer learning in three steps: we took a DNN that had already been trained on ImageNet and then on ChestX-ray14, and we applied the third step, training it on the COVID-19 dataset.

But we can also train other neural networks with this twice transfer learning if, after downloading them pretrained on ImageNet, we train them on ChestX-ray14 and then on the COVID-19 dataset. Looking for twice transfer learning on other works, we found that it was already used, with success, for mammogram images classification (Cai et al. (2018)).

Output neuron keeping

In this paper, we propose an original addition to the twice transfer learning technique: output neuron keeping. In three-step transfer learning, we look for a task in the second step that is very similar to the final step task. Having two alike datasets, one might find that they share classes. In our study, ChestX-ray14 and the COVID-19 dataset have both a class for healthy individuals (called "no findings" in ChestX-ray14 and "normal" in the COVID-19 dataset). Also, ChestX-ray14 shares the pneumonia class with our dataset.

Thus, we suggest that, having the same or very similar classes in the second and third step of twice transfer learning, when preparing the network for the third step, one could keep the output neurons that classify those classes and change only the other output neurons. Doing this, the representations that these artificial neurons learned in the second step can be maintained and this may improve training speed or performance in the final task.

To keep output neurons, a simple approach in PyTorch begins by copying their weights and biases at the end of twice transfer learning step two (training on the second dataset). Then, in the beginning of step 3, change the DNN output layer to match the new desired output format, find the neurons that will classify the classes similar to step two's, and substitute their weights and biases for the copied ones.

Trained DNNs

In this work, we trained five DNNs, which we will call A, B, C, D, and E.

Network A is a 201 layers DenseNet, downloaded pretrained on ImageNet, and trained again on the COVID-19 dataset. Thus, it received a simple transfer learning approach.

Network B is also a DenseNet201, downloaded pretrained on ImageNet. But it was then trained on ChestX-ray14 and then on the COVID-19 dataset. Thus, it used twice transfer learning (with ImageNet in the first step, ChestX-ray14 in the middle step and the COVID-19 dataset in the last).

Network C is the same as network B, but, besides the twice transfer learning approach, we used output neuron keeping: the neurons that classified the no findings and pneumonia classes in ChestX-ray14 were assigned to classify normal and pneumonia in the COVID-19 dataset, the other output neurons were removed and a new one, with random weights and biases, was added to classify the chance of COVID-19. Thus, here we used a twice transfer learning with output neuron keeping.

Network D is a 121-layer DenseNet. We downloaded a pretrained CheXNet (already trained on ImageNet and then on ChestX-ray14) and trained it on the COVID-19 database. So, it had a twice transfer learning, but only the last step was done by us.

The last network, E, is also a DenseNet121. It began as a pretrained CheXNet and we again trained it on the COVID-19 dataset. But, before training on this dataset, we copied the weights and bias from the neuron that classified pneumonia to the one that would classify pneumonia on the final database. Thus, it used a twice transfer learning with output neuron keeping, and maintained just one output neuron. We note that CheXNet had no neuron to classify the chances of healthy lungs (it had 14 output neurons, one for each of the 14 diseases on ChestX-ray14), therefore, we could not keep two output neurons.

Our motivation to choose working with dense neural networks was CheXNet's excellent result in ChestX-ray14, even surpassing 4 radiologists in pneumonia detection (Rajpurkar et al.(2017)).

Table 1 summarizes all DNNs we created.

Data processing and augmentation

ChestX-ray14

As we would also train DenseNets in this dataset, we based our dataset processing for ChestX-ray14 in what the authors did when training CheXNet (Rajpurkar et al. (2017)). All images were resized to 224×224 size, with 3 channels, and normalized with the mean and standard deviation from the network's previous training in ImageNet. We used the originally reported test dataset as our test dataset. The other images were randomly split, with 80% of the X-rays to create the training dataset and 20% for validation (holdout). Different datasets had no images from the same patient. Fifteen labels

Table 1 Created deep neural networks

Name	DNN architecture	Training process
A	201-layers DenseNet	Transfer learning
B	201-layers DenseNet	Twice transfer learning
C	201-layers DenseNet	Twice transfer learning + output neuron keeping
D	121-layers DenseNet (CheXNet)	Twice transfer learning
E	121-layers DenseNet (CheXNet)	Twice transfer learning + output neuron keeping

were created, one for each disease and one for "no findings" (they were organized in a binary vector with 15 dimensions).

Like the authors did when training CheXNet (Rajpurkar et al. (2017)), we applied random horizontal flips (with 50% chance) to the training images before giving them to the DNN. This was done online and, if the image was flipped, we would only feed the new image to the network, not the new and the old one (thus, not making the mini-batch bigger).

COVID-19

Firstly, we divided our assembled COVID-19 dataset into three: training, validation and test. To create the test dataset, we randomly took 50 images of each class (normal, pneumonia and COVID-19). After removing the 150 test images, we took 90% of the remaining images for training and 10% for validating. This was also done randomly, but preserving the same class proportions in the two datasets. Again, different datasets had no images from the same patient. All images were loaded in greyscale (to minimize color variations between the datasets), reshaped from 1024×1024 pixels to 224×224, converted to three channels and then normalized to the same mean and standard deviation used in the ChestX-ray14 and ImageNet normalization.

The CheXpert database had 8 pneumonia images that were labeled by three board-certified radiologists. These images were included in our testing dataset, along with 42 other random pneumonia images from the CheXpert.

Many of the images had letters or words on them, and some of these words were exclusive for certain classes. For example, some COVID-19 images (from Italy) had the word "SEDUTO" (Italian word for "seated") written on the upper left corner. We were afraid that this could affect the network classification performance; hence, we decided to manually edit our test dataset images, removing the words or letters. They were simply covered with black rectangles and, as they were not over the lungs, no relevant information was lost. The objective was only to test the network ability analyzing the lungs and, by editing only the test dataset, there would be no risk of teaching the DNN to identify our black rectangles during training.

We decided to apply data augmentation for two reasons: it improves the DNN performance for small datasets (like our COVID-19 database), and because it would balance our datasets. As we would also benefit from a balanced validation dataset, we applied augmentation in training and validation.

We used three image augmentation methods: rotations (between −40 and 40 degrees), translations (up to 28 pixels left and right or up and down) and flipping (horizontal). These transformations could augment our data and also make the DNN more robust to input translations and rotations. All augmentation was done online and, after one operation, we would not substitute the original image, we would just add the new one, randomly rotated, translated and possibly flipped (50% chance), to the mini-batch (making it bigger). We augmented our normal image database 30 times, our pneumonia images 8 times, and our COVID-19 images 24 times. We ended up with a training dataset of 8280 COVID-19, 8640 pneumonia and 8640 normal lung augmented images. To feed the DNN a completely balanced dataset, 360 pneumonia and normal lung augmented images were left out in each epoch. In every epoch we randomly changed which images would be left out, therefore every image was used in the training process.

Creating and training the DNNs

On the ChestX-ray14 dataset

As networks B and C have the same training process in the twice transfer learning first and second steps, we are able to train only one network on the ChestX-ray14 database, which would be used for creating networks B and C in the future.

To create the DNN we downloaded an ImageNet pretrained PyTorch version of DenseNet201. The only changes we made on it was substituting its output layer for one with 15 neurons (one for each of the 14 diseases in this dataset and one for the "no findings" class) and we kept this layer's activation as a sigmoid function. The training process was carried out in PyTorch, with binary cross entropy loss, stochastic gradient descent with momentum of 0.9, mini-batches of 16 images, and hold-out validation. We trained the networks on two NVidia GTX 1080 GPUs.

We began by freezing all model parameters except for the output layer's and training for 20 epochs, with a learning rate of 0.001. We then set the learning rate to 0.0001, unfroze all

the model parameters and trained for 90 epochs more, in the end of which the DNN was already overfitting.

On the COVID-19 dataset

To create network A we downloaded the ImageNet pretrained PyTorch version of DenseNet201, removed the output layer and added a new one, with three neurons and softmax activation. For DNN B we started with the neural network we had trained on ChestX-ray14, also removed its last layer and added a new one, like the above mentioned. For network C we made the same output layer substitution in the network we had trained on ChestX-ray14, but we copied the weights and biases from the output neurons that classified "no findings" and "pneumonia" to the ones that would classify "normal" and "pneumonia" in the COVID-19 dataset.

For network D, we downloaded a pretrained CheXNet (on ImageNet and ChestX-ray14) from Zech (2018), and proceeded by also changing its final layer for one with three neurons and softmax activation. Finally, for network E, we downloaded the same network (Zech (2018)), also changed the final layer as in network D, but we copied the weights and bias for the output neuron that classified pneumonia to the one that would again classify pneumonia, in the new dataset.

For all networks, the training process in the COVID-19 dataset was the same, given that their architectures were similar, and this allowed us to better compare the transfer learning methods. We used PyTorch, cross entropy loss, stochastic gradient descent with momentum of 0.9, and mini-batches of 9 images. We trained the networks on two NVidia GTX 1080 GPUs. We also used holdout validation. Most training parameters were determined with many preliminary tests, for example, weight decay was used in the beginning to avoid fast overfitting, but was removed when we noticed the DNNs had stopped improving training error.

The training process had 4 phases, which we will describe now. We began by freezing all the network parameters except for the output layer's and we trained for 10 epochs, using a learning rate of 0.001 and weight decay of 0.01. We then unfroze all parameters, trained for 48 epochs, with early stop and patience of 20, learning rate of 0.0001 in the last layer and decreasing by a factor of 10 for each previous dense block and its transition layer. We lowered the learning rate to 0.00001 (now in every layer) and trained for 48 epochs more, with the same early stop and weight decay. Finally, for the last phase, we removed the weight decay and early stop and trained for 48 epochs again.

Layer-wise relevance propagation

Layer-wise Relevance Propagation (LRP) is an explanation technique that aims to make DNNs (complex and nonlinear structures with millions of parameters and connections) interpretable by humans. It decomposes the network prediction, showing, in a heatmap, how each input variable contributed to the output (Bach et al. (2015)). We note that interpreting deep neural networks can be challenging. A Taylor Decomposition, based on the Taylor expansion of the network output, is unstable in deep neural networks, due to noisy gradients and the existence of adversarial examples (Montavon et al. (2019)).

We may choose any output neuron to start the relevance propagation, and this choice will define the meaning of the colors in the resulting heatmap. For example, if we choose to start LRP on the output neuron that classifies COVID-19, red colors on the map will indicate areas that the DNN associated with COVID-19, while blue areas will point characteristics associated with the other classes, normal and pneumonia. Examples can be seen in the section "Analysis with LRP." Unless stated otherwise, the heatmaps shown in this paper were created starting the propagation at the winning output neuron, i.e., the one with the highest output for the input X-ray.

LRP is based on propagating the DNN prediction backward through the layers, using local propagation rules, which may change for different layers. This relevance propagation has a conservation property, in the way that the quantity of relevance a neuron receives from the upper layer will be distributed in equal amount to the neurons in the lower layer (Montavon et al. (2019)). This property ensures that the quantity of explanation we get in the input (in the heatmap) relates to what can be explained by the output. As examples of medical contexts in which LRP was used we can cite neuroimaging (Thomas et al. (2019)) and explaining therapy predictions (Yang et al. (2018)).

Analyzing our network with LRP allows us to identify problems in the DNN classification method, and also to generate a heatmap of the X-ray image, showing where in the lungs the network identified issues. This map could be given to radiologists along with the network predictions, helping them to verify the classifier analysis, providing insights about the X-rays and allowing a more profitable cooperation between human experts and artificial intelligence.

We can choose between many propagation rules in each neural network layer, and presets are selections of these rules for the many layers in a DNN. We can compare them, searching for one that creates good human interpretability and fidelity to the network operation. We used the Python library iNNvestigate (Alber et al. (2019)), which already implemented LRP for DNNs like DenseNet and has parameter presets that work well for these networks. This library works with Keras and TensorFlow, but we trained our DNNs on PyTorch, thus, we used the library pytorch2keras (Malivenko (2018)) to convert our models. After the conversion, we tested them again, and obtained the same accuracies

we had on PyTorch, confirming that the conversion worked well.

Results

Figure 1 shows how test accuracy changed during training on the COVID-19 dataset, for all five DNNs. These measurements were taken for the best networks (according to validation loss) in each of the four training phases described in the section "Creating and Training the DNNs" (which correspond to epochs 10, 58, 106 and 154). Our best test accuracy was 100%, achieved by both the CheXNets and the DenseNet201 with twice transfer learning and output neuron keeping.

In Tables 2 and 3, we show the confusion matrices from networks that did not achieve 100% accuracies, DNNs A and B. Network B made 2 mistakes in the 150 test images: 2 COVID-19 images were classified as pneumonia. DNN A misclassified 1 healthy patient as COVID-19.

With 100% accuracy, our DNNs C, D and E have precision, recall and F1 score of 1 in our test dataset.

We applied Layer-wise Relevance Propagation to the trained neural networks. The generated heatmaps allowed us to analyze how each part of the input X-rays influenced the DNN classification. This topic will be discussed in more detail in the section "Analysis with LRP," along with examples of the heatmaps.

Discussion

Analyzing Fig. 1, we start comparing the DenseNet201 DNNs. In the first epochs, we note that network C, with twice transfer learning and output neuron keeping, started with noticeably better accuracy, followed by network B and then network C, which used simple transfer learning. The performance differences become smaller with more training. When we compare the best results, we observe that DNN C had 100% accuracy, DNN A 99.3% and DNN B 98.7%.

Comparing the two CheXNets, we observe that their performances were similar; both had the same accuracies in the beginning and achieved equal results in the end. We think that output neuron keeping with only one neuron had a smaller effect than with two neurons, therefore the difference between networks B and C are larger than between DNNs D and E. We also see that these networks started with accuracies higher than network A and B, but smaller than DNN C, which used output neuron keeping on two neurons.

Analyzing Tables 2 and 3, we observe that even on the networks that committed mistakes, no patient with some disease was classified as healthy, which would be the most dangerous type of misclassification.

We achieved test accuracies of 100% with three networks, but we must note that our test dataset only has 150 images. Maybe these DNNs could make some mistakes in a bigger test database (which we did not use due to the limitation in the number of COVID-19 images).

Looking at the state-of-the-art in COVID-19 detection with deep learning, we observe that, according to the review

Table 2 Confusion matrix for network A

		Predicted class		
		Normal	Pneumonia	COVID-19
Real class	Normal	49	0	1
	Pneumonia	0	50	0
	COVID-19	0	0	50

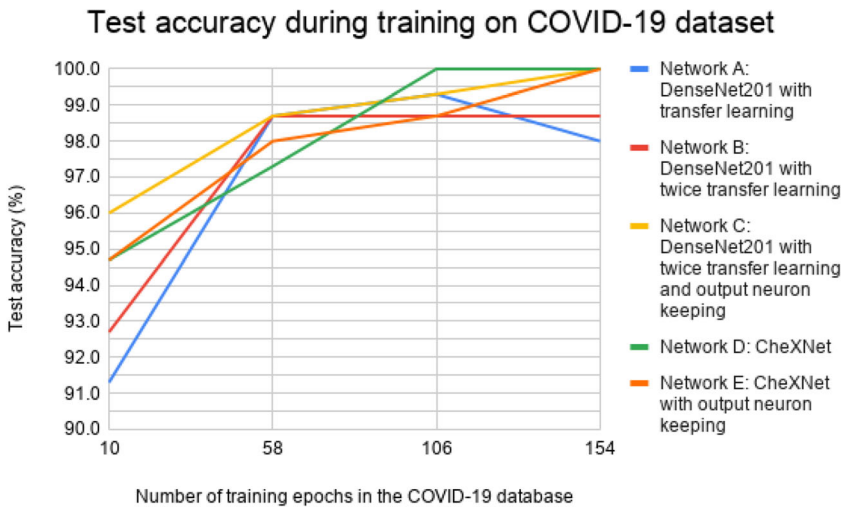

Fig. 1 Test accuracies during training plot

Table 3 Confusion matrix for network B

		Predicted class		
		Normal	Pneumonia	COVID-19
Real class	Normal	50	0	0
	Pneumonia	0	50	0
	COVID-19	0	2	48

Shoeibi et al. (2020), most of the techniques using DNNs have accuracies in the 90% to 100% range. Therefore, our work is on par with the current state-of-the-art.

Analysis with LRP

We tested different LRP presets on iNNvestigate and got more understandable and coherent heatmaps with "LRP-PresetAFlat." Figure 2 shows a correctly classified COVID-19 X-ray test image and the heatmap for it, taken from one of our best performing DNNs, network C. The more red the region on the map, the more important it was for the DNN classification as COVID-19. The more blue, the more that region is related to other classes (like a healthy part of the lung or a region that the DNN assosciated to pneumonia sympthoms). We observed our DNN found signs of COVID-19 in both lungs. We also note that, on some images, our black rectangles can create some artifacts in their borders (red or blue lines, as can be seen on the neck region of Fig. 4), but, given that they were used only for testing and in all classes, we do not think this can affect accuracy.

We decided to analyze the effect of words and letters on the X-ray images, fearing that words used on the datasets could be sources of bias. We used the same test COVID-19 image shown in Fig. 2, but without removing the word "SEDUTO" from its upper right corner and the letters "DX" from the upper left corner. The resulting heatmap is shown in Fig. 3 (also created with network C). It becomes clear, by the red color on the map, that the network learned to associate these words with the COVID-19 class.

To measure the effect of this problem we tested the DNNs we trained with our testing dataset but unedited (with the words and letters that it originally had). This test generated only small changes in accuracy, which increased or decreased, at most, 1.33% on fully trained networks. DNNs on early training stages showed changes up to 3.33%. Our best neural networks (C, D and E), when fully trained, showed no accuracy change.

Another test was trying to "fool" our networks, adding to a COVID-19 lungs X-ray test image the letters "L PA," which were copied from a healthy image (L above and PA below, in a small rectangle). The network C given probability for COVID-19 changed from 99.94 to 99.91%, and for normal increased from 0.036 to 0.055%. Figure 4 shows a heatmap created by LRP starting on the output neuron that classifies the normal class. Therefore, red colors indicate areas associated with normal. We can observe that the letters PA are red on the heatmap, meaning that they are influencing the DNN to choose normal instead of COVID-19. The letter L, above the PA, is blue. This letter was very common in all classes, thus it was not associated with the normal class. Given the small output probabilities change, we see that, in this case, the letters effect was tiny.

Conclusion

The proposed method of output neuron keeping, with twice transfer learning, outperformed the sole use of twice transfer learning and simple transfer learning in the 201-layer dense networks. Taking into account this work and the great results three steps transfer learning had in Cai et al. (2018), we think that the technique and our output neuron keeping method are promising and could also improve performances in other classification problems.

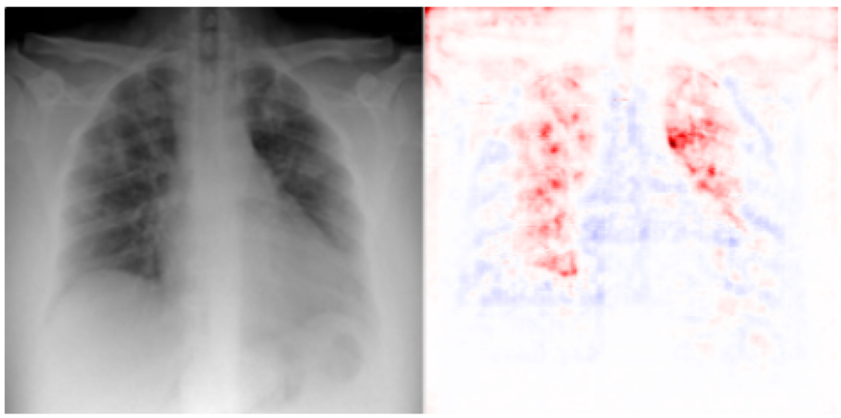

Fig. 2 Heatmap for test COVID-19 image

Fig. 3 Heatmap for test COVID-19 image without removing word "SEDUTO" and letters "DX"

We were surprised by the fact that the CheXNet DNNs could keep up with and even surpass most of the DenseNet201 DNNs. We concluded that the main reason for this is that, even though we used data augmentation, the effect of overfitting was stronger on the 201-layer dense neural networks.

LRP showed promising results highlighting details in the X-rays that most influenced the network classification. We hope that this may indicate a possibility to help radiologists and provide a better interaction between experts and artificial intelligence. It also allowed us to discover that words and letters can influence the DNN classifications. This influence was small in the fully trained DNNs, with accuracies changing at most 1.33% if we do not remove the words and letters from the testing dataset. On DNNs at the beginning of the training process this effect was larger, with accuracies increasing or decreasing up to 3.33%.

We should state that the dataset used in this study is not ideal. Although we used the largest open COVID-19 X-ray database that we could find in October 2020, we could only utilize 439 coronavirus X-rays. A much larger dataset, with all the classes collected from the same sources, would allow the creation of better generalizing classification models. It would also reduce possible causes of bias, as different sources can have different characteristics, like the letters and words that we analyzed in this work. We emphasize the need for such a database and hope to continue this research when it becomes available. Furthermore, clinical studies would be required to ensure that the high accuracies obtained in this and other studies (Shoeibi et al. (2020)) would also be achieved in a real-world scenario.

Although larger databases and clinical studies are still needed, this study and other initiatives (Wang and Wong (2020), Shoeibi et al. (2020)) show that DNNs have the potential of making chest X-ray a fast, accurate, cheap and easily available auxiliary method for COVID-19 diagnosis. The trained networks proposed here are open source and available for download in Bassi and Attux (2020): we hope DNNs can be further tested in clinical studies and help in the creation of tools to fight the COVID-19 pandemic.

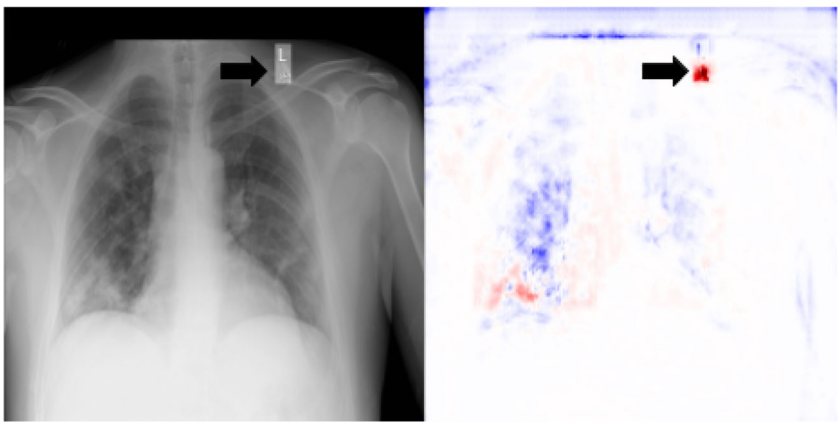

Fig. 4 Normal output neuron heatmap for test COVID-19 image edited with letters (indicated by arrows) copied from a normal image

Funding This work was partially supported by CNPq (process 308811/2019-4) and CAPES.

Declarations

Conflict of interest The authors declare no competing interests.

References

Alber M, Lapuschkin S, Seegerer P, Hägele M, Schütt KT, Montavon G, et al. innvestigate neural networks! J Mach Learn Res. 2019;20:1–8.

Bach S, Binder A, Montavon G, Klauschen F, Müller KR, Samek W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS One. 2015;10:1–46.

Bassi, P.R.A.S., Attux, R. Covid-19 twice transfer dnns. 2020. https://github.com/PedroRASB/COVID-19-Twice-Transfer-DNNs. Accessed 16 Jun 2020.

Bengio Y, Courville A, Vincent P. Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell. 2012;35(8):1798–828.

Cai Q, Liu X, Guo Z. Identifying architectural distortion in mammogram images via a se-densenet model and twice transfer learning. 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI). 2018;1–6.

Cohen JP, Morrison P, Dao L. Covid-19 image data collection. arXiv2003.11597. 2020.

Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. ImageNet: a large-scale hierarchical image database. CVPR09. 2009.

Goodfellow I, Bengio Y, Courville A, Bengio Y. Deep learning.volume 1. Cambridge: MIT press; 2016.

Hopkins UJ. Coronavirus Resource Center. 2020. https://coronavirus.jhu.edu/. Accessed 16 Jun 2020.

Huang G, Liu Z, van der Maaten L, Weinberger KQ. Densely connected convolutional networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016;2261–2269.

Irvin J, Rajpurkar P, Ko M, Yu Y, Ciurea-Ilcus S, Chute C, et al. Chexpert: a large chest radiograph dataset with uncertainty labels and expert comparison: arXiv:1901.07031; 2019.

Jaeger S, Candemir S, Antani S, Wang J, Lu PX, Thoma G. Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. Quant Imaging Med Surg. 2014;4:475–7. https://doi.org/10.3978/j.issn.2223-4292.2014.11.20.

Malivenko G. Pytorch2keras. 2018. https://github.com/nerox8664/pytorch2keras. Accessed 16 Jun 2020.

Montavon G, Binder A, Lapuschkin S, Samek W, Müller, Klaus-Robert EW, et al. Layer-wise relevance propagation: an overview. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning: Springer International Publishing; 2019. p. 193–209.

Rajpurkar P, Irvin J, Zhu K, Yang B, Mehta H, Duan T, et al. Chexnet: radiologist-level pneumonia detection on chest x-rays with deep learning: CoRR abs/1711.05225; 2017.

Shoeibi A, Khodatars M, Alizadehsani R, Ghassemi N, Jafari M, Moridian P, et al. Automated detection and forecasting of covid-19 using deep learning techniques: a review: ArXiv abs/2007.10785; 2020.

Thomas AW, Heekeren HR, Müller KR, Samek W. Analyzing neuroimaging data through recurrent deep learning models. Front Neurosci. 2019;13:1321.

Trunk GV. A problem of dimensionality: a simple example. IEEE Trans Pattern Anal Mach Intell. 1979;1:306–7.

Wang X, Peng Y, Lu L, Lu Z, Bagheri M, Summers RM. Chestx-ray8: hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. IEEE Conf Comput Vis Pattern Recogn. 2017;2017:3462–71.

Wang L, Wong A. Covid-net: a tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images: arXiv:2003.09871; 2020.

Wang W, Xu Y, Gao R, Lu R, Han K, Wu G. Detection of sars-cov-2 in different types of clinical specimens. JAMA. 2020.

Yang Y, Tresp V, Wunderle M, Fasching PA. Explaining therapypredictions with layer-wise relevance propagation in neural networks. IEEE Int Conf Healthc Inform. 2018;2018:152–62.

Zech J. Reproduce-chexnet. 2018. https://github.com/jrzech/reproduce-chexnet. Accessed 16 Jun 2020.