A Deep Convolutional Neural Network for COVID-19 Detection Using Chest X-Rays

Pedro R. A. S. Bassi, Romis Attux

Department of Computer Engineering and Industrial Automation, School of Electrical and Computer Engineering, University of Campinas - Campinas, Brazil
p157007@dac.unicamp.br, ORCID: 0000-0002-8995-9423, attux@dca.fee.unicamp.br

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, viral 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 an output neuron keeping strategy. 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 state-of-the-art test accuracy (99.4%) and F1 Score (0.994). Twice transfer learning improved classification performances in the first half of the training process but, after many epochs, we could achieve the same results using common transfer learning. Output neuron keeping improved twice transfer learning performances. Although LRP showed that words on the X-Rays can influence the networks predictions, we discovered this had only a very small effect on accuracy.

Conclusion: 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.

Keywords:
COVID-19 Detection, Neural Networks, Chest X-Ray, LRP, Twice Transfer Learning, Output neuron keeping

1. Introduction

In 2020, COVID-19 became pandemic, affecting both developed and developing countries around the world. By 06/16/2020, the virus had already infected more than 8,000,000 people and caused more than 438,000 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 (Chowdhury et al. (2020), Wang and Wong (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. In Chowdhury et al. (2020), a larger COVID-19 dataset was reported, and the authors had a maximum of 98.3% test accuracy, with 96.7% recall and 100% precision regarding SARSCoV-2 (Severe Acute Respiratory Syndrome Coronavirus 2).

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 to differentiate between normal lungs, COVID-19 and viral pneumonia, using an open COVID-19 X-Ray dataset assembled in Chowdhury et al. (2020). Other DNN is a 201 layers DenseNet that had been pretrained on ImageNet (Deng et al. (2009)) and we also fine tuned it in the 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 this 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 the smaller dataset containing the COVID-19 class (Chowdhury et al. (2020)).

In this paper, we also suggest a new 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 third database (from Chowdhury et al. (2020)) 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, viral 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. LRP allows us to have a better understanding of the DNN operation, but 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).

2. Databases

2.1. NIH ChestX-Ray14

ChestX-Ray14 is one of the largest chest X-ray datasets, with 112,120 images from 30,805 patients. 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 was originally labeled with Natural Language Processing in the image associated radiological reports and the accuracy of the labels is estimated to be 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.

2.2. COVID-19 Database

In this study, we used the open dataset reported in Chowdhury et al. (2020). The database is composed of 219 COVID-19 chest X-Ray images,

as well as 1341 normal lung images and 1345 viral pneumonia images. It is available on Kaggle and is one of the largest open collections of COVID-19 X-Rays to date.

As described in Chowdhury et al. (2020), this dataset was created with images of chest X-Ray. The COVID-19 data was taken from different databases: 63 images from the Italian Society of Medical and Interventional Radiology (SIRM) COVID-19 DATABASE (SIRM (2020)), 60 from the Novel Corona Virus 2019 Dataset (Cohen et al. (2020)) and 60 images were collected (by the authors of Chowdhury et al. (2020)) from 43 recently published articles. The normal and viral pneumonia images were taken from the Kaggle database Chest X-Ray images (pneumonia) (Kermany et al. (2018)). More information about the dataset can be found in Chowdhury et al. (2020).

We modified this dataset by removing all lateral-view X-Rays (there were only 13, 5.9% of the COVID-19 images). As this view was exclusive to the COVID-19 class, we considered these images could teach our networks to associate all lateral view X-Rays to SARSCoV-2 and also act as noise during training. Although these images were present in the dataset released by the authors of Chowdhury et al. (2020) on Kaggle, we note that they also used only frontal views in their study.

3. Transfer Learning, Twice Transfer Learning and Output Neuron Keeping

3.1. Introduction to transfer learning

When we add dimensions to a neural network input, the data tends to becomes sparser. Thus, with larger inputs (like our 224x224 images), we need more data to create a good statistical model of the inputs and labels distribution. This problem is known as "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 avoiding 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.

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. 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.

3.2. Twice transfer learning and output neuron keeping

It is common to choose ImageNet as the first dataset (Chowdhury et al. (2020), Rajpurkar et al. (2017)) when we have image classification tasks, 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, viral 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 (Deng et al. (2009)), then on ChestX-Ray14 (Wang et al. (2017)) and, finally, on the COVID-19 dataset (Chowdhury et al. (2020)). 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 (Cai et al. (2018)).

In this paper, we propose a new 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 has the pneumonia class, that is very similar to the "viral pneumonia" we have in the final database. 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 an output neuron, a simple approach in PyTorch begins by copying its 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.

4. 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 viral 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 viral pneumonia. 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 to keep (it had 14 output neurons, one for each of the 14 diseases on ChestX-Ray14).

Our motivation to choose working with dense neural networks was CheXNet excellent result in ChestX-Ray14, even surpassing 4 radiologists in pneumonia detection (Rajpurkar et al. (2017)).

Table 1 summarizes all DNNs we created.

5. Data Processing and Augmentation

5.1. 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

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

Table 1: DNNs

(Rajpurkar et al. (2017)). All images were resized to 224x224 size, with 3 channels, normalized with the mean and standard deviation from the network's previous training in ImageNet. We used 70% of the images to create the training dataset, 20% for test and 10% for validation (holdout). Different datasets had no images from the same patient. 15 labels 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 batch bigger).

5.2. COVID-19

Firstly, we divided the dataset from Chowdhury et al. (2020) into three: training, validation and test. To create the test dataset, we randomly took 60 images of each class (normal, viral pneumonia and COVID-19). The test dataset in Chowdhury et al. (2020) has the same configuration. After removing the 180 test images, we took 80% of the remaining images for training and 20% for validating. This was also done randomly, but preserving the same class proportions in the two datasets. All images were reshaped from 1024x1024 pixels to 224x224, with 3 channels.

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 SIRM (2020)) 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 idea 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, as the authors found out in Chowdhury et al. (2020)), 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 40 pixels left and right and up to 28 pixels 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 to the batch (making it bigger). We augmented our normal and viral pneumonia image database 8 times (2 random rotations, followed by 2 random translations and flipping), and our COVID-19 images 72 times (6 random rotations, then 6 random translations and flipping). We ended up with a training dataset of 8409 COVID-19, 8128 viral pneumonia and 8128 normal lung images.

6. Creating and Training the DNNs

6.1. On the ChestX-Ray14 dataset

As networks B and C have the same training process in the twice transfer learning first and second steps, so 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 a 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.

6.2. On the COVID-19 dataset

To create network A we downloaded the pretrained PyTorch version of DenseNet201, removed the output layer and added a new one, with three neurons and sigmoid 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, alike 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 "viral 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 sigmoid 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 classify "viral 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 allowing us to better compare the transfer learning methods. We used PyTorch, binary 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 learning rate of 0.001 and weight decay of 0.01. We then unfroze all parameters, set the learning rate to 0.0001 and trained for 48 epochs, with early stop and patience of 20. We lowered the learning rate to 0.00001 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.

7. Layer-wise Relevance Propagation

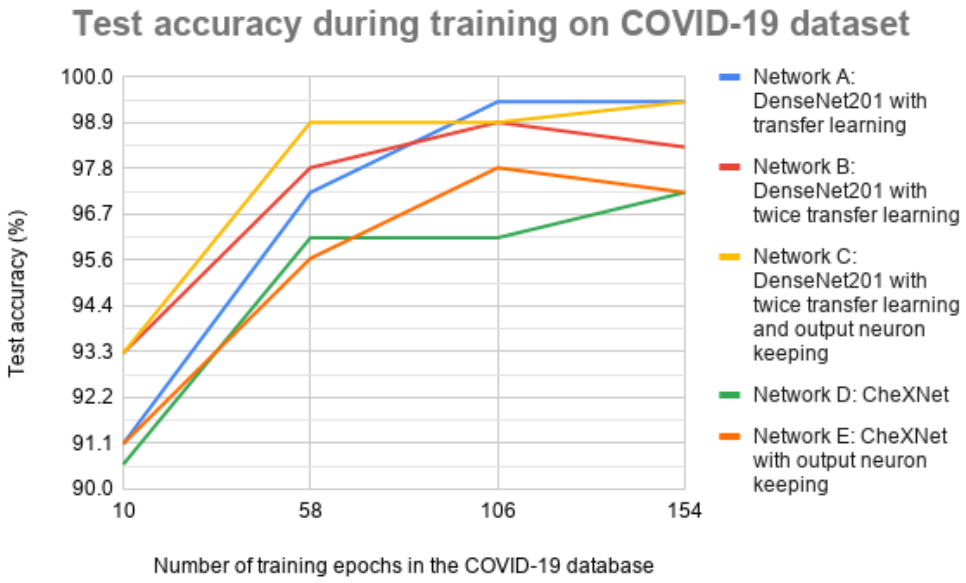
Layer-wise Relevance Propagation (LRP) is an explanation technique that aims to make DNNs (complex and nonlinear structures with millions of parameters and connections) interpretable by humans. It decomposes the network prediction, showing, in a heatmap, how each input variable contributed to the output (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)).

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 lower layers (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 allow 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

Figure 1: Test accuracies during training plot.



(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.

8. Results

Figure 1 shows how test accuracy changed during training on the COVID-19 dataset, for all five DNNs. This measurements were taken for the best networks (acording to validation loss) in each of the four training phases described in section "Creating and Training the DNNs" (which correspond to epochs 10, 58, 106 and 154). Our best test accuracy was 99.4%, achieved by both the 201 layers dense neural networks trained using common transfer learning and using twice transfer learning with output neuron keeping (networks A and C).

We now analyze the confusion matrices from the two networks that had 99.4% test accuracy after training (A and C). We note that they were equal, thus, we can show them in Table 2. The networks made only one mistake in the 180 test images: one COVID-19 image was classified as normal lungs. In Table 3, we show network metrics for these DNNs.

		Predicted class		
		Normal	Viral Pneumonia	COVID-19
Real Class	Normal	60	0	0
	Viral Pneumonia	0	60	0
	COVID-19	1	0	59

Table 2: Confusion matrix for the two best neural networks (B and C)

Class	Recall	Precision	F1 Score
Normal	1	0.984	0.992
Viral Pneumonia	1	1	1
COVID-19	0.983	1	0.991
All classes mean	0.994	0.995	0.994

Table 3: Network metrics for the 2 best neural networks (B and C)

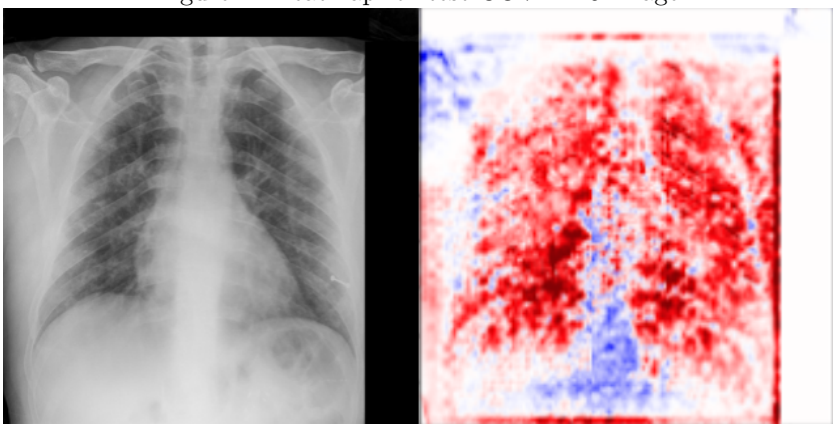
We applied Layer-wise Relevance Propagation to the trained neural networks. The generated heatmaps allowed us to analyze how each part of the input X-Rays influenced the DNN classification. This topic will be discussed in more detail in section "Analysis with LRP", along with examples of the heatmaps.

9. Discussion

Analyzing Figure 1 we observe that networks using twice transfer learning and output neuron keeping ended up with better results than their counterparts that used only twice transfer learning (networks C and E outperformed B and D, respectively). Another conclusion was that dense neural networks with 201 layers performed better than the ones with 121 layers. At last, comparing network A with networks B and C, we see that twice transfer learning improved the performances in the beginning of training, but with many epochs DNN A was able to obtain the same performance.

We can compare our results to those obtained by the networks the authors trained in Chowdhury et al. (2020), with the same database (albeit with a different treatment). Although they had expressive results, networks A and C outperformed all DNNs from the paper (AlexNet, ResNet18, DenseNet201 and SqueezeNet) in accuracy and F1 Score. Their F1 Scores were 0.971 for a

Figure 2: Heatmap for test COVID-19 image



DenseNet201 with data augmentation and 0.983 for a SqueezeNet, also with data augmentation.

9.1. Analysis with LRP

We tested different LRP presets on iNNvestigate and got more understandable and better heatmaps with "LRP-PresetAFlat". Figure 2 shows a correctly classified COVID-19 X-Ray test image and the heatmap for it, taken from network C. The more red the region on the map, the more it was important 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). We observe our DNN found signs of COVID-19 all over the lungs, but mostly lower in the right lung. We also note that our black rectangles created some artifacts (red lines) in their borders, but, given that they were used only in testing and in all classes, we do not think this can affect accuracy.

We decided to analyze the effect of words on the X-Ray images. We used the same test COVID-19 image shown in figure 2, but without removing the word "SEDUTO" from its upper right corner. The resulting heatmap is shown in Figure 3 (also for network C). It becomes clear, by the red color on the map, that the network learned to associate this word with the COVID-19 class.

To measure the effect of this problem we tested DNNs we trained with our testing dataset but unedited (with the words and letters it originally had). This did not change the accuracies of our fully trained networks, but had a small effect on networks that were not completely trained yet. For example, a

Figure 3: Heatmap for test COVID-19 image without removing word “SEDUTO”

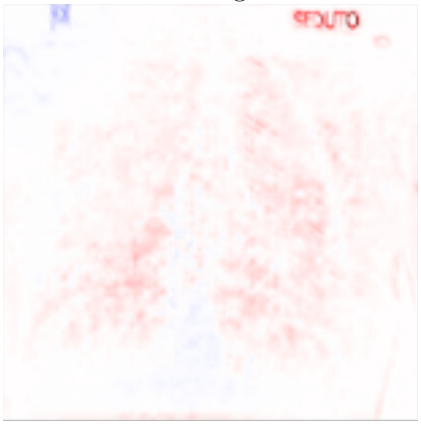
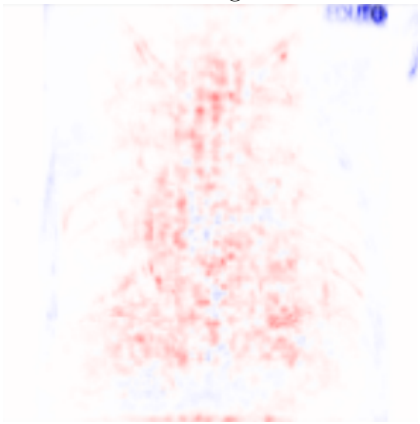


Figure 4: Heatmap for test Normal image edited with the word “SEDUTO”



CheXNet with about 94% test accuracy had its accuracy increased in almost 1% and we could also see accuracies decreasing slightly.

Another test was to try to “fool” our networks, adding to a normal lungs X-Ray test image that was classified correctly the word “SEDUTO”. Interestingly, the network C given probability for Normal just changed from 91.8% to 91.6%, and for COVID-19 increased from 0.456% to 0.713%. Figure 4 shows that the word influenced negatively the DNN decision for the normal class (as it is blue), but, given the small output change, we see the effect was tiny.

10. Conclusion

The proposed method of twice transfer learning and output neuron keeping outperformed the sole use of twice transfer learning both with the 201-layer dense networks and with the 121-layer ones. Also, with an analysis of the DenseNets201, we observe that twice transfer learning generated better performance than the simple transfer learning until about half of the training process, but at the end this two methods gave the same performance. 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 could improve performances in other classification problems.

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 slightly influence the DNN classifications.

With 99.4% test accuracy and F1 Score our classifier is a state-of-the-art DNN for classifying COVID-19 with chest X-Rays. A performance comparable to that of an expert had already been achieved by deep networks in pneumonia classification using radiography (Rajpurkar et al. (2017)). This study and other initiatives (Wang and Wong (2020), Chowdhury 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 network proposed here is 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.

11. Acknowledgments

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

References

Alber, M., Lapuschkin, S., Seegerer, P., Hägele, M., Schütt, K.T., Montavon, G., Samek, W., Müller, K.R., Dähne, S., Kindermans, P.J., 2019. innvestigate neural networks! Journal of Machine Learning Research 20, 1–8. URL: http://jmlr.org/papers/v20/18-540.html.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Mller, K.R., Samek, W., 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLOS ONE 10, 1–46. URL: https://doi.org/10.1371/journal.pone.0130140, doi:10.1371/journal.pone.0130140.

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

Bengio, Y., Courville, A., Vincent, P., 2012. Representation learning: A review and new perspectives. arXiv:1206.5538.

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

Chowdhury, M.E.H., Rahman, T., Khandakar, A., Mazhar, R., Kadir, M.A., Mahbub, Z.B., Islam, K.R., Khan, M.S., Iqbal, A., Al-Emadi, N., Reaz, M.B.I., 2020. Can ai help in screening viral and covid-19 pneumonia? arXiv:2003.13145.

Cohen, J.P., Morrison, P., Dao, L., 2020. Covid-19 image data collection. arXiv 2003.11597 URL: https://github.com/ieee8023/covid-chestxray-dataset.

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. ImageNet: A Large-Scale Hierarchical Image Database, in: CVPR09.

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

Hopkins, U.J., 2020. Coronavirus resource center. URL: https://coronavirus.jhu.edu/.

Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q., 2016. Densely connected convolutional networks. arXiv:1608.06993.

Kermany, D., Goldbaum, M., Cai, W., Valentim, C., Liang, H.Y., Baxter, S., McKeown, A., Yang, G., Wu, X., Yan, F., Dong, J., Prasadha, M., Pei, J., Ting, M., Zhu, J., Li, C., Hewett, S., Dong, J., Ziyar, I., Zhang, K., 2018.

Identifying medical diagnoses and treatable diseases by image-based deep learning. Cell 172, 1122–1131.e9. doi:10.1016/j.cell.2018.02.010.

Malivenko, G., 2018. pytorch2keras. URL: https://github.com/nerox8664/pytorch2keras.

Montavon, G., Binder, A., Lapuschkin, S., Samek, W., Müller, Klaus-Robert", e.W., Montavon, G., Vedaldi, A., Hansen, L.K., Müller, K.R., 2019. Layer-wise relevance propagation: An overview, in: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. Springer International Publishing, pp. 193–209.

Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D.Y., Bagul, A., Langlotz, C., Shpanskaya, K.S., Lungren, M.P., Ng, A.Y., 2017. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. CoRR abs/1711.05225. URL: http://arxiv.org/abs/1711.05225, arXiv:1711.05225.

SIRM, S.I.D.R.M.e.I., 2020. Covid-19 database. URL: https://www.sirm.org/category/senza-categoria/covid-19/.

Thomas, A.W., Heekeren, H.R., Müller, K.R., Samek, W., 2019. Analyzing neuroimaging data through recurrent deep learning models. Frontiers in Neuroscience 13, 1321. URL: http://dx.doi.org/10.3389/fnins.2019.01321, doi:10.3389/fnins.2019.01321.

Trunk, G.V., 1979. A problem of dimensionality: A simple example. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-1, 306–307.

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

Wang, W., Xu, Y., Gao, R., Lu, R., Han, K., Wu, G., 2020. Detection of sars-cov-2 in different types of clinical specimens. JAMA doi:10.1001/jama.2020.3786.

Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., Summers, R.M., 2017. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on

weakly-supervised classification and localization of common thorax diseases, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3462–3471.

Yang, Y., Tresp, V., Wunderle, M., Fasching, P.A., 2018. Explaining therapy predictions with layer-wise relevance propagation in neural networks, in: 2018 IEEE International Conference on Healthcare Informatics (ICHI), pp. 152–162.

Zech, J., 2018. reproduce-chexnet. URL: `https://github.com/jrzech/reproduce-chexnet`.