Screening CT Images for COVID-19 Infections and Pneumonia

Lorenzo Loconte
University of Bari Aldo Moro
l.loconte5@studenti.uniba.it
744328

Giuseppe Colavito
University of Bari Aldo Moro
g.colavito2@studenti.uniba.it
736047

I. Introduction

The goal of this work is to understand if a patient has a COVID-19 infection or a non-COVID-19 pneumonia, given some CT (computed tomography) images of the lungs. The task is a multi-class classification task with three classes: *normal*, *pneumonia* and *COVID-19*. In this kind of task, it is important to don't have *false-negatives*, since an infected patient can have several health problems and can help the infection spreading.

In this work two approaches are used to solve the problem, both in the area of deep learning. The first is to classify each image separately, hence producing several predictions for a single CT scan. The second is to classify a CT scan, consisting of several CT images, in an end-to-end fashion. Moreover, attention mechanisms are used to generate some explanations regarding the predictions, a very important aspect in the underlying medical domain.

II. Dataset

The COVIDx-CT dataset is composed of 143,778 training examples, 25,486 validation examples and 25,658 testing examples. Each example is a CT image (i.e. a slice of a CT scan) and it is annotated with one between three labels: *normal* (normal CT image), *pneumonia* (pneumonia not caused by COVID-19) and *covid19* (pneumonia caused by COVID-19). However, all CT slices belonging to a single CT scan have the same label. In other words, we also have annotated CT scans, consisting of several CT slices. It's important to notice that the three classes (*normal*, *pneumonia* and *covid19*) are highly unbalanced. Moreover, within the dataset, the bounding box coordinates for of lungs region are also available for each example.

The class distributions are available in Table I. The dataset is freely available on Kaggle.

TABLE I
COVIDx-CT examples distribution for each class.

COVIDx-CT	**Normal**	**Pneumonia**	**COVID-19**	**Total**
Train	35,996	25,496	82,286	143,778
Validation	11,842	7400	6244	25,486
Test	12,245	7395	6018	25,658

Furthermore, another dataset called COVIDx-SeqCT is obtained from the original dataset. The COVIDx-SeqCT dataset is composed of 2390 training examples, 430 validation examples and 408 test examples. Each example consists of a sequence of exactly 16 CT slices extracted uniformly from CT scans. CT scans in the original dataset having less than 16 slices are discarded. All the sequences of CT images are annotated with one between three labels: *normal* (normal CT image), *pneumonia* (pneumonia not caused by COVID-19) and *covid19* (pneumonia caused by COVID-19). The class distributions of COVIDx-SeqCT are available in Table II.

TABLE II
COVIDx-SeqCT examples distribution for each class.

COVIDx-SeqCT	**Normal**	**Pneumonia**	**COVID-19**	**Total**
Train	486	403	1501	2390
Validation	172	112	146	430
Test	174	100	134	408

Figure 1 shows four CT images for each class and Figure 2 shows a CT sequence consisting of 16 CT images.

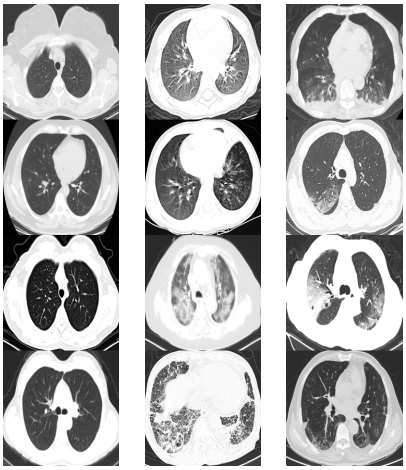

Fig. 1. COVIDx-CT dataset samples. Normal CT images (on the left), non-COVID-19 pneumonia CT images (at the center) and COVID-19 pneumonia CT images (on the right).

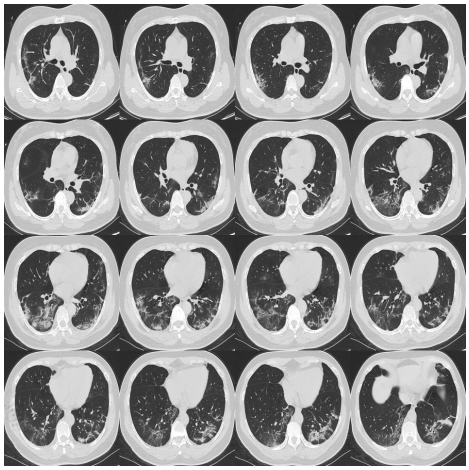

Fig. 2. A CT scan (from left to right and top to bottom) consisting of 16 CT images taken from COVIDx-SeqCT.

A. Preprocessing

The dataset is preprocessed with cropping and resizing. Only the part of the image inside the bounding box is kept. A resizing step is necessary to have all the images with the same size and to fit the input size requirements of the neural networks we will use for the task. So, all the images are resized to 224×224 using bicubic interpolation.

III. Attention Mechanisms

In this section we are going to introduce some basic attention techniques that will be used in the next sections of this work, especially for the task of images and sequences classification. Most importantly, attention techniques were used to produce attention maps (showed in appendix) for both images and sequences classification.

In deep learning, attention can be interpreted as a vector of importance weights. We estimate using the attention vector how much a region of an image, or a part of a sequence, is correlated with (*attends to*) other elements. Those elements can be other image regions, parts of an output sequence and also a predicted class.

Particularly, attention mechanisms have been effectively used in neural machine translation [1] and neural image caption generation [2]. Furthermore, some attention techniques permits to introduce some kind of explainability in deep learning models, especially in deep convolutional neural networks (CNNs).

Typically, there are two kinds of attentions: *soft* and *hard*. In soft attention techniques, alignment weights are learned over all patches in the input data. In hard attention techniques, alignments weights are learned singularly over only one patch at a time. Soft attention modules are easy to integrate in deep neural networks since they are smooth and differentiable. However, hard attention modules are typically non-differentiable and more difficult to integrate in deep neural networks. For more details, please refer to [2].

Attention mechanisms are also divided in two categories: *global* and *local*. Global attention modules are similar to soft attention modules. Local attention modules are a combination of both soft and hard attention techniques. Usually, local attention modules improves a hard attention technique by making it smooth and differentiable. For more details, please refer to [3].

A. Attention for Sequence Classification

Now, we formalize a simple global/smooth attention mechanism, derived from Bahdanau et al. [1] and Luong et al. [3] adapted for sequence classification. Formally, let $\boldsymbol{x}$ be a source sequence of length n. Moreover, given a bidirectional recurrent neural network (RNN) with a forward hidden state $\overrightarrow{\boldsymbol{h}}_i$ and a backward hidden state $\overleftarrow{\boldsymbol{h}}_i$ at position i, a simple concatenation of them represents a hidden state $\boldsymbol{h}_i$ at position i.

Let $\boldsymbol{h}_n$ be the last hidden state given by the RNN. A linear transformation with $\tanh$ activation function is used to obtain the global feature vector $\boldsymbol{g}$.

$$\boldsymbol{g} = \tanh(\boldsymbol{W}_g \boldsymbol{h}_n + \boldsymbol{b}_g) \tag{1}$$

The resulting feature vector $\widehat{\boldsymbol{g}}$, called *context vector* is obtained by the attention module. Formally, the attention module computes the following equations.

$$s_i = \boldsymbol{g}^T \boldsymbol{W}_s \boldsymbol{h}_i \tag{2}$$

$$a_i = \frac{\exp s_i}{\sum_{j=1}^{n} \exp s_j} \tag{3}$$

$$\widehat{\boldsymbol{g}} = \sum_{i=1}^{n} a_i \boldsymbol{h}_i \tag{4}$$

The quantities s_i and a_i are called *scores* and *attentions* respectively. So, the context vector is a linear combination of hidden states $\boldsymbol{h}_i$ with attentions a_i, such that $\sum_i^n a_i = 1$. Finally, a probability distribution over the classes $\boldsymbol{c} = \{c_1, \ldots, c_k\}$ is obtained by a dense layer with softmax activation function.

$$\boldsymbol{v} = \boldsymbol{W}_c \widehat{\boldsymbol{g}} + \boldsymbol{b}_c \tag{5}$$

$$p(c_i) = \frac{\exp v_i}{\sum_{j=1}^{k} \exp v_j} \tag{6}$$

The parameters of the attention module can be trained jointly with the rest of the model.

B. Attention for Image Classification

For images classification, another global/soft attention mechanism was introduced by Jetley et al. [4], consisting of several attention modules that can be introduced in state-of-the-art convolutional neural networks such as VGG and ResNet. The main idea is to use attention maps to identify and exploit spatial information used by CNNs in making their classification decisions, while suppressing irrelevant information of other regions. Basically, the attention module *enforces* a notion of compatibility between local features that are extracted at intermediate layers in a CNN and global

features that are fed directly to a linear classifier at the end of a CNN.

Formally, let $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \dots & \boldsymbol{x}_n \end{bmatrix}$ be the set of feature vectors extracted at a certain convolutional layer. Each $\boldsymbol{x}_i$ is the vector of output activations at the spatial location i. The *global feature* vector $\boldsymbol{g}$ is the output of the network's sequence of convolutional and non-linear layers, having only to pass through the fully connected block to produce the prediction. The method proceeds by projecting the set of feature vectors $\boldsymbol{X}$ into the vector space of $\boldsymbol{g}$ using a linear mapping. Then the set of *compatibility scores* is computed.

$$\widehat{\boldsymbol{X}} = \boldsymbol{W}_p \boldsymbol{X} \tag{7}$$

$$s_i = \mathcal{C}(\widehat{\boldsymbol{x}}_i, \boldsymbol{g}) \tag{8}$$

Here, s_i are the compatibility scores between the projected features vectors $\widehat{\boldsymbol{x}}_i$ and the global feature vector $\boldsymbol{g}$. Then the compatibility scores are normalized by a softmax operation, hence obtaining the *attentions*. Similarly to the attention technique already introduced for sequence classification, the new global features $\widehat{\boldsymbol{g}}$ is obtained as a linear combination of feature vectors $\boldsymbol{x}_i$ with attentions a_i such that $\sum_{i=1}^{n} a_i = 1$.

$$a_i = \frac{\exp s_i}{\sum_{j=1}^{n} \exp s_j} \tag{9}$$

$$\widehat{\boldsymbol{g}} = \sum_{i=1}^{n} a_i \boldsymbol{x}_i \tag{10}$$

The compatibility score function $\mathcal{C}$ can be defined in various ways. The *alignment model* proposed by Bahdanau et al. [1] can be used as a compatibility function as follows.

$$\mathcal{C}(\widehat{\boldsymbol{x}}_i, \boldsymbol{g}) = \langle \boldsymbol{u}, \widehat{\boldsymbol{x}}_i + \boldsymbol{g} \rangle \tag{11}$$

The weight vector $\boldsymbol{u}$ can be interpreted as learning the set of features that are actually relevant. Alternatively, we can use the dot product as a simpler measure of their compatibility as follows.

$$\mathcal{C}(\widehat{\boldsymbol{x}}_i, \boldsymbol{g}) = \langle \widehat{\boldsymbol{x}}_i, \boldsymbol{g} \rangle \tag{12}$$

In this case, the magnitude of the scores are directly proportional to the alignment between $\widehat{\boldsymbol{x}}_i$ and $\boldsymbol{g}$. Empirical results showed that the compatibility function in Equation (11) generally outperforms the one in Equation (12).

In general, multiple attention modules are used in CNNs at different depth, hence obtaining a set of global features vector $G = \begin{bmatrix} \widehat{\boldsymbol{g}}_1 & \dots & \widehat{\boldsymbol{g}}_m \end{bmatrix}$ where m is the number of attention modules. Then the global features computed by the attention modules can be combined either using concatenation or by feeding them into multiple linear classifiers and averaging the predicted classes, as in well known bagging techniques.

As for the attention technique for sequence classification, the parameters of the attention module can be trained jointly with the rest of the model.

IV. First approach: single image classification

The first approach we tried, consists in classifying every single image separately. This means that, if a patient has multiple images, it will have multiple classifications. This is the simplest approach, but has some problems: what happens if a patient with multiple images has one of his images classified as *pneumonia* and another one as *covid19*?

Furthermore, every image of the sequence has the same label, since the patient's health state is evaluated by an expert considering all of the images. However, a single image may be not enough to classify the patient. So this approach may seem a bit naive.

To avoid this problem, we have also tried a sequence based method, to have a single classification for the whole sequence and so, a single classification for the patient. We will talk about this approach on the next section. We have also focused our work on how to get attention maps and therefore explainable results.

A. ResNet50 wtih Attention

ResNets [5] is a class of CNNs that is a special case of *residual networks*. A residual network is a network in which, instead of approximating a mapping $H(x)$, we let the network approximate a residual function $F(x) := H(x) - x$. This formulation is used to address the *degradation problem*. With the network depth increasing, the accuracy gets saturated and then degrades rapidly. A ResNet architecture uses 3×3 convolutional layers with batch normalization [6] and ends with a global average pooling (GAP) layer and a fully-connected layer with softmax activation function. Shortcut connections are inserted. The identity shortcuts can be directly used when the input and output are of the same dimensions. When the dimension increase, an identity shortcut with 0-padding or a projection shortcut can be used.

Figure 3 shows two kinds of residual convolutional blocks based on residual connections: *basic* and *bottleneck*. Bottleneck residual blocks differ from basic residual blocks for the fact that it is composed by 1×1 convolutions. For this reason, bottleneck residual blocks are more efficient and therefore permitting to better scale deeper residual architectures.

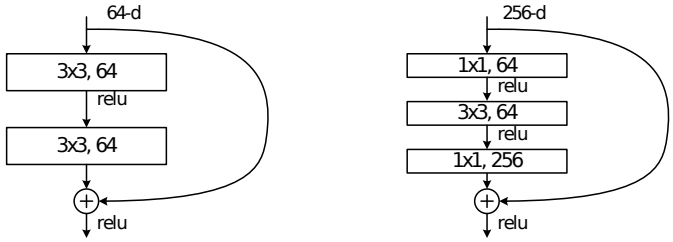

Fig. 3. Illustration of both a basic residual block (on the left) and a bottleneck residual block (on the right).

We trained a slightly modified version of ResNet50 by adding a tail of three additional bottleneck residual blocks, having the same number of channels of a ResNet50 features extractor. Hence, the total number of convolutional layers is 59. Furthermore, two attention modules for image classification have been introduced after convolutional layers number

40 and 49, as discussed in Section III-B. The first attention module process features maps of size $1024 \times 14 \times 14$ and the second attention module process features maps of size $2048 \times 7 \times 7$. Therefore the attention maps given by the two attention modules are of size 14×14 and 7×7 respectively and normalized by softmax.

The compatibility score function $\mathcal{C}$ used for both the attention modules is the parametrized compatibility score written in Equation (11). The global features computed by the attention modules are combined by concatenation, hence obtaining a vector embedding of size 3072. Finally, a fully connected layer with softmax activation function is used as linear classifier.

We will refer to this model as RESNET50-ATT2.

B. Experimental Setting

Data augmentation has been used because, in general, having augmented data can help to achieve a lower generalization error and higher classification accuracy, as shown in [7], [8]. The dataset is augmented using random horizontal and vertical flip, gaussian blur with kernel size 7 and standard deviation sampled uniformly from $[0.05, 2.0]$. Moreover, random affine transformations including random scaling (with scale sampled uniformly from $[0.9, 1.1]$), random rotation (with degrees sampled uniformly from $[-30, 30]$) and random translation (with translation percentages sampled uniformly from $[0.0, 0.1]$) are introduced as well. Moreover, random shear augmentation on the x-axis with rotation sampled uniformly from $[-20, 20]$ is used.

The default layers of RESNET50-ATT2 are initialized with pretraining on *ImageNet*. The Adam optimizer [9] with learning rate of 5×10^{-4} and other parameters to their default values are used. The batch size is set to 64. For each epoch, 1000 optimization steps are made. Furthermore, in order to mitigate the problem of unbalanced training data, a *weighted binary cross-entropy* is used as the loss function. The weights of the loss function are inversely proportional with respect to the class frequencies in the training data. The training process is stopped either after 25 epochs or if no improvements on the validation loss happen after 5 consecutive epochs.

C. Evaluation

The model has been evaluated on the test set with precision, recall and F_1 metrics on all the three classes. Table III shows the performance metrics on the three classes separately and the weighted average metrics.

TABLE III
PERFORMANCE METRICS FOR EACH CLASS USING RESNET50-ATT2.

Class	Precision	Recall	F_1
Normal	0.992	0.992	0.986
Pneumonia	0.967	0.963	0.965
COVID-19	0.936	0.960	0.947
Weighted Average	0.971	0.971	0.971

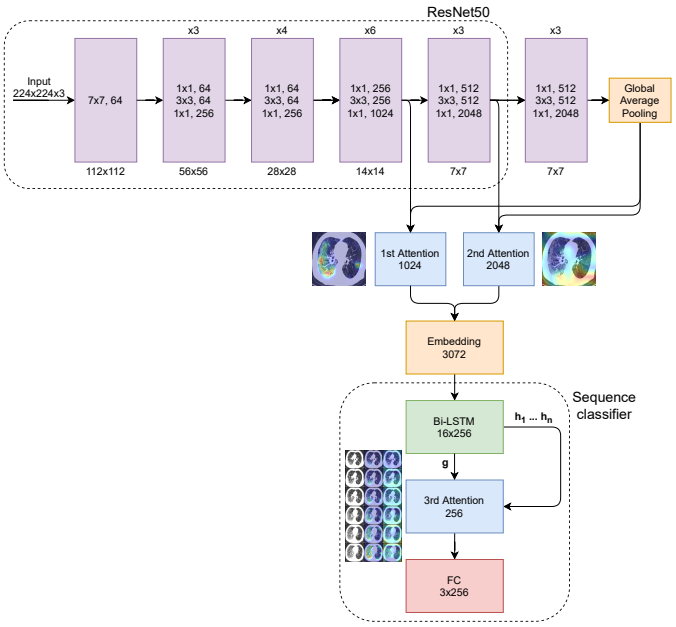

Fig. 4. The complete model architecture consisting of a RESNET50-ATT2 image embeddings extractor and a BI-LSTM with soft attention for sequence classification, as discussed in Section V.

V. SECOND APPROACH: SEQUENCE BASED CLASSIFICATION

The second approach consists in using sequences of images instead of single images. To do so, we used a Bidirectional LSTM model. The idea behind this choice is that the CT-images are a sequence and they need to be classified as a single example. In this way, a single patient will have a single label, instead of multiple predicted labels for each image of the CT sequence.

Using this approach means also to use a fixed sequence length, hence some of the sequences are discarded from the dataset since they don't have enough images. Another problem arises when the sequences have more images then the fixed sequence size, so also in this case, some examples are discarded. This also makes the results of the two models harder to compare, since it's like using two different datasets.

So, the main idea consists of using a CNN, like a RESNET50, as an image embeddings extractor and successively applying a RNN, like a LSTM, with a linear classifier. The joint model is also called CNN-LSTM and was proposed by Nguyen et al. [10] for the task of CT images sequence classification.

In this work this technique is adapted accordingly in order to incorporate attention modules in both the image embeddings extractor and the sequence classifier. We will refer to this model with RESNET50-LSTM-ATT2, consisting of a RESNET50-ATT2 as an image embeddings extractor and a BI-LSTM with soft attention and a linear classifier for end-to-end CT images sequence classification. Figure 4 shows the full architecture of the proposed model.

A. LSTM with Soft Attention

A long short-term memory network is a type of recurrent neural network (RNN). Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video).

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. This is essentially what LSTM does: it saves information for later, thus preventing older signals from gradually vanishing during processing.

LSTM recurrent networks have "LSTM cells" that have an internal recurrence (a self-loop), in addition to the outer recurrence of the RNN. Each cell has the same inputs and outputs as an ordinary recurrent network, but has more parameters and a system of gating units that controls the flow of information. The most important component is the state unit $s_i^{(t)}$ (where i is the cell and t is the time step) that has a linear self-loop. The self-loop weight is controlled by a forget gate unit $f_i^{(t)}$, that sets this weight to a value between 0 and 1 via a sigmoid unit:

$$f_i^{(t)} = \sigma\left(b_i^f + U_{i,*}^f x^{(t)} + W_{i,*}^f h^{(t-1)}\right) \tag{13}$$

where $x^{(t)}$ is the current input vector and $h^{(t)}$ is the current hidden vector, containing the output of all the LSTM cells, and b^f, U^f, W^f are respectively biases, input weights and recurrent weights for the forget gates. $U_{i,*}$ denotes the i-th row of U and $W_{i,*}$ denotes the i-th row of W. The LSTM cell internal state is thus updated as follows, but with a conditional self-loop weight $f_i^{(t)}$:

$$s_i^{(t)} = f_i^{(t)} s_i^{(t-1)} + g_i^{(t)} \sigma\left(b_i + U_{i,*} x^{(t)} + W_{i,*} h^{(t-1)}\right) \tag{14}$$

where b, U, and W respectively denote the biases, input weights and recurrent weights into the LSTM cell.

The external input gate unit $g_i^{(t)}$ is computed similarly to the forget gate but with its own parameters:

$$g_i^{(t)} = \sigma\left(b_i^g + U_{i,*}^g x^{(t)} + W_{i,*}^g h^{(t-1)}\right) \tag{15}$$

The output $h_i(t)$ of the LSTM cell can also be shut off via the output gate $q_i^{(t)}$, which also uses a sigmoid unit for gating:

$$h_i^{(t)} = \tanh\left(s_i^{(t)}\right) q_i^{(t)} \tag{16}$$

$$q_i^{(t)} = \sigma\left(b_i^o + U_{i,*}^o x^{(t)} + W_{i,*}^o h^{(t-1)}\right) \tag{17}$$

which has parameters b^o, U^o, and W^o for its the biases, input weights and recurrent weights respectively. LSTM networks have been shown to learn long-term dependencies more easily than the simple recurrent architectures [11].

A Bidirectional LSTM (also called BI-LSTM) consists of two LSTM layers that operate in opposite directions: *forward* and *backward*. The hidden states of both directions are usually concatenated to obtain a single hidden state for each element of the sequence.

Furthermore, a soft attention module for sequence classification has been added after the BI-LSTM module, as discussed in Section III-A. This permits us to obtain an attention map that encodes the importance of each feature in the sequence for the classification.

B. Experimental Setting

The vector embeddings for each CT image are extracted from a RESNET50-ATT2 pretrained from the previous experiment on single CT image classification. The weights of the CT image embeddings extractor are kept frozen.

The data augmentation used is the same as in the case of the RESNET50-ATT2 training, but it's applied to the whole sequence. The Adam optimizer [9] with learning rate of 1×10^{-4} and other parameters to their default values are used. The batch size is set to 16 and L_2 weight decay have been introduced with a factor of 5×10^{-3}. The loss function used is again the *weighted binary cross-entropy*, as in RESNET50-ATT2. The training process is stopped either after 50 epochs or if no improvements on the validation loss happen after 10 consecutive epochs.

C. Evaluation

The model has been evaluated on the test set with precision, recall and F_1 metrics on all the three classes. Table IV shows the performance metrics on the three classes separately and the weighted average metrics.

TABLE IV
PERFORMANCE METRICS WITH RESNET50-LSTM-ATT2 MODEL.

Class	Precision	Recall	F_1
Normal	1.000	0.983	0.991
Pneumonia	1.000	0.980	0.990
COVID-19	0.964	1.000	0.982
Weighted Average	0.988	0.988	0.988

VI. CONCLUSIONS AND FUTURE WORKS

We proposed two different approaches, one based on single image classification and one on sequence classification. Both of the approaches are performing quite well, but they are not directly comparable, since they use different datasets. Furthermore, the COVIDx-SeqCT is very small, so the sequence-based model should be tested on a bigger dataset to assess better the performances. We think that the sequence-based approach is more correct then evaluating single images separately, that looks naive. Since a patient has more images for a single CT-exam, a patient is being classified n times and it is difficult to understand how to get a single classification for the patient from the n classifications of his CT images. Instead, evaluating a patient from the whole sequence seems more natural and may give a better overview of the patient's health state.

It could be interesting to test more complex attention mechanisms and compare the heatmaps, trying to understand the most relevant features for the CT images.

In appendix, several attention maps are showed in Figures 5 and 6. Moreover, as showed in Figure 7, attention maps can be combined in order to get segmentation masks of relevant regions.

REFERENCES

[1] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2015.

[2] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, 2015.

[3] M. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *CoRR*, 2015.

[4] S. Jetley, N. A. Lord, N. Lee, and P. H. S. Torr, "Learn to pay attention," *CoRR*, vol. abs/1804.02391, 2018.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[6] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 448–456. [Online]. Available: http://proceedings.mlr.press/v37/ioffe15.html

[7] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," *ArXiv*, vol. abs/1712.04621, 2017.

[8] I. Sirazitdinov, M. Kholiavchenko, R. Kuleev, and B. Ibragimov, "Data augmentation for chest pathologies classification," *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pp. 1216–1219, 2019.

[9] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014.

[10] N. T. Nguyen, D. Q. Tran, N. T. Nguyen, and H. Q. Nguyen, "A cnn-lstm architecture for detection of intracranial hemorrhage on CT scans," *CoRR*, vol. abs/2005.10992, 2020.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

APPENDIX
ATTENTION MAPS AND SEGMENTATION MASKS VISUALIZATION

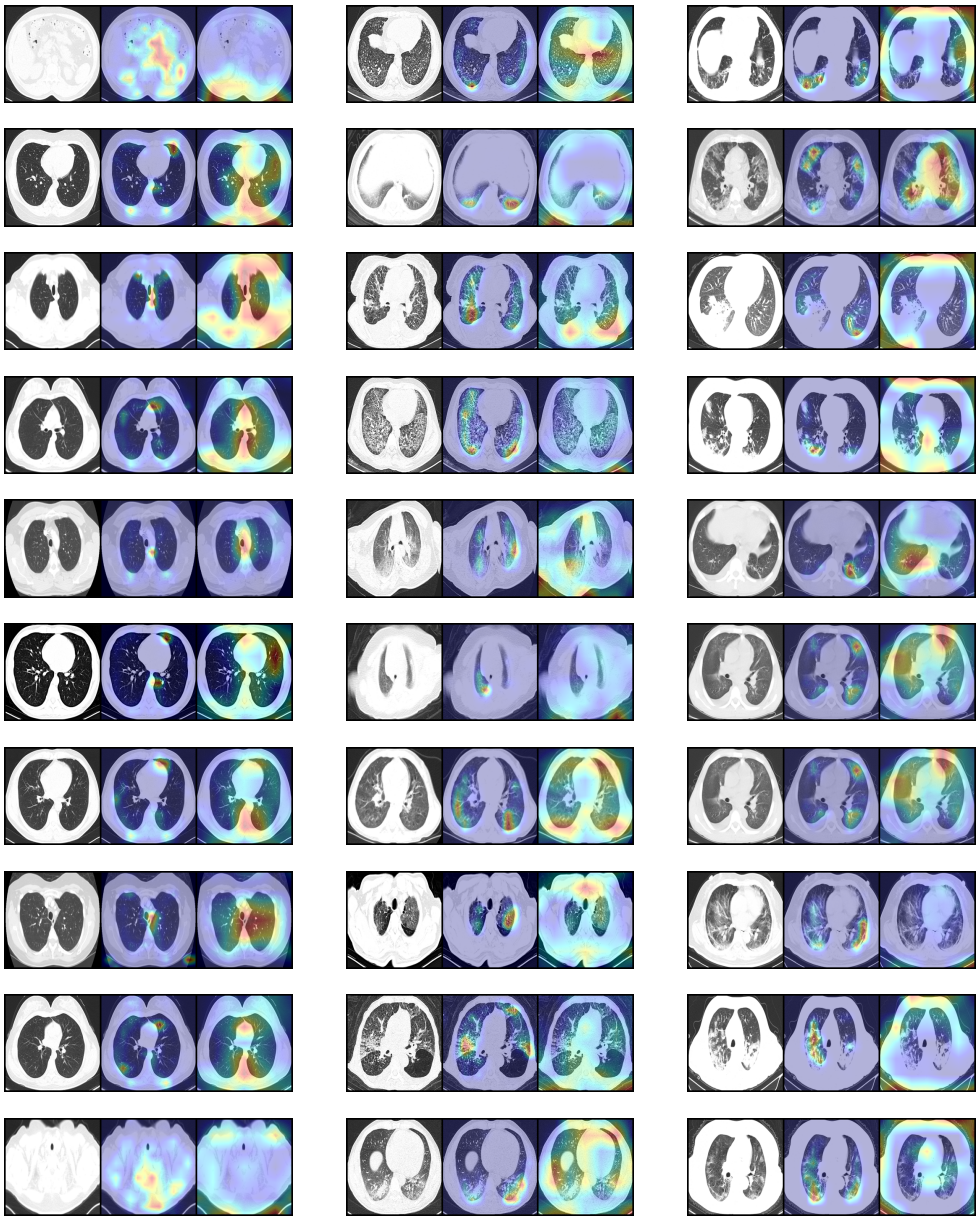

Fig. 5. Attention maps given from RESNET50-ATT2. Normal CT images (on the left), non-COVID-19 pneumonia CT images (at the center) and COVID-19 pneumonia CT images (on the right).

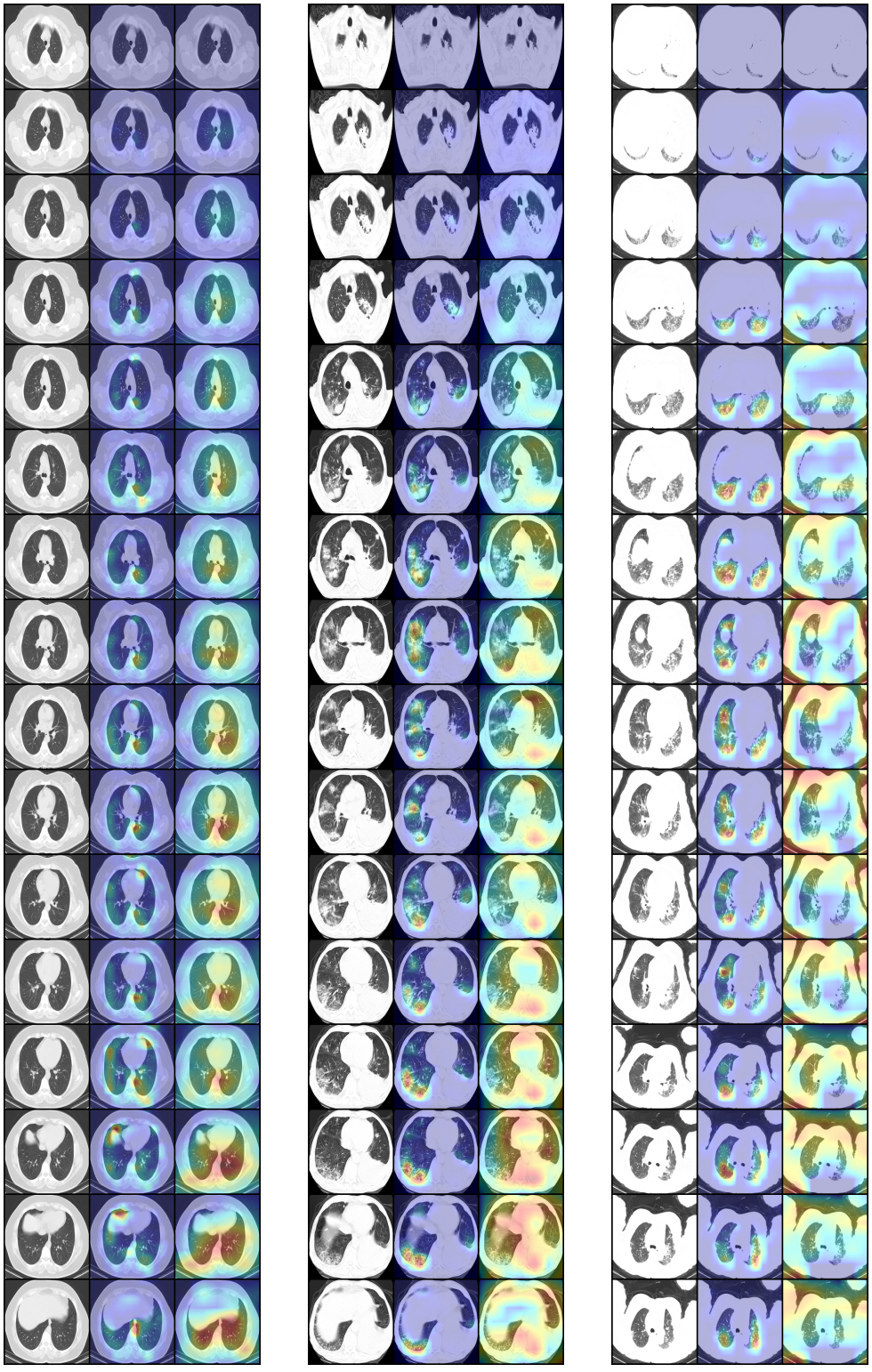

Fig. 6. Attention maps given from RESNET50-LSTM-ATT2. The intensity of the attention maps have been weighted according to the sequence attention map given by BI-LSTM. Normal CT scan images (on the left), non-COVID-19 pneumonia CT scan images (at the center) and COVID-19 pneumonia CT scan images (on the right).

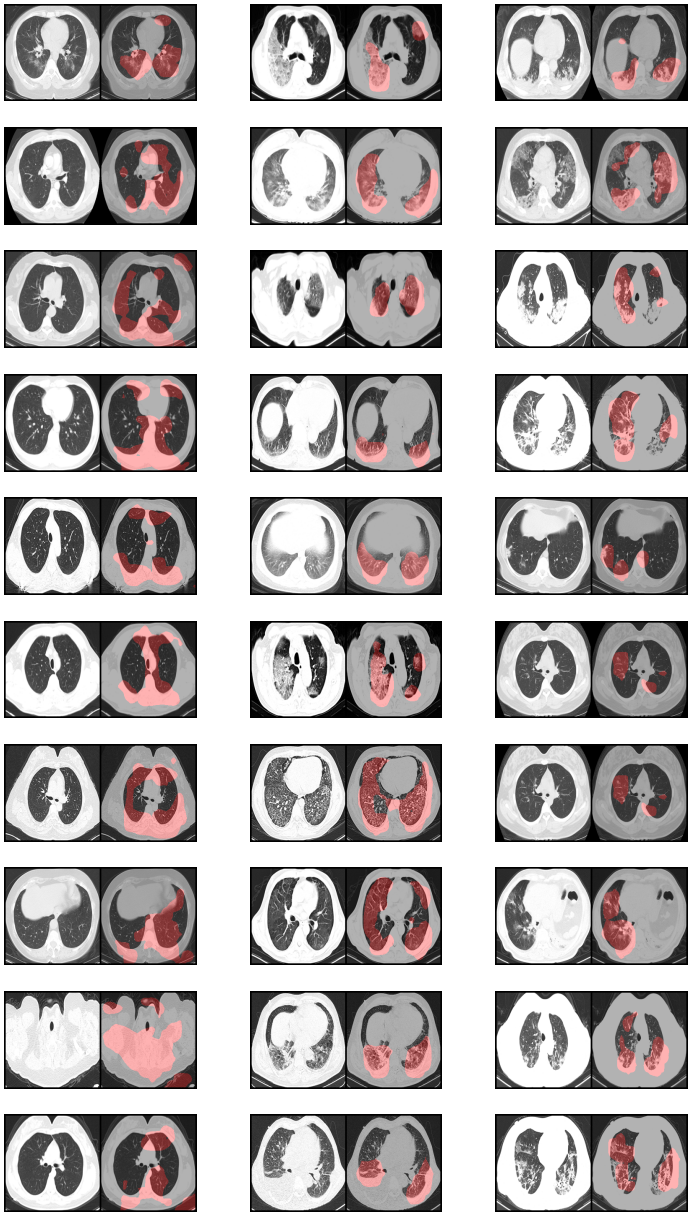

Fig. 7. Segmentation masks given from RESNET50-ATT2, obtained by combining the attention maps (with a square-rooted element-wise product) and then binarizing the results using the OTZU method. Normal CT images (on the left), non-COVID-19 pneumonia CT images (at the center) and COVID-19 pneumonia CT images (on the right).