nature portfolio

https://doi.org/10.1038/s42256-022-00483-7

Supplementary information

An interpretable deep learning workflow for discovering subvisual abnormalities in CT scans of COVID-19 inpatients and survivors

In the format provided by the authors and unedited

Supplementary Information

Content

List of Supplementary Figures

List of Supplementary Tables

List of Supplementary Algorithms

1 Technical Details for the DLPE Method

The DLPE method depends on fast, robust and accurate segmentations of various chest tissues. Supplementary Sections 1.1-1.4 describe the technical details, model performance and the model analysis of these segmentation models.

1.1 CT data normalization

Our CT datasets are collected from 5 different hospitals. Thus, the imaging parameters and the ground truth annotations vary greatly among different centers.

Our previous study [1] gave an effective normalization procedure that cast any chest CT into a machine-agnostic standard embedding space, which significantly improved the robustness and accuracy when segmenting COVID-19 lesions. In this study, we follow the key idea of our previous normalization procedure, while making the procedure more intuitive.

1.1.1 CT dataset quality control (QC)

Dataset of good quality is an essential prerequisite for developing state-of-the-art machine learning models. Thus, it is crucial to delete cases with wrong ground truth annotations, impaired CT data or CT data with extreme noise. In general, we use 3D visualizations, 2D visualizations as well as statistical methods to ensure that the CT data and the ground truth annotations are of good quality.

QC for the CT data: we use the following steps to check whether a CT data is of good quality.

Step 1): use the state-of-the-art model in our previous study [1] to get the lung segmentation.

Step 2): 3D visualize the lung segmentation by stereolithography (STL). Check whether the segmentation looks natural (explained in Supplementary Section 1.2.9). If there exist obvious defects, mark the CT data quality as "risky".

Step 3): "smoothness" analysis of the lung segmentation. Let b to be the number of boundary voxels of the lung mask, and v be the number of lung voxels, then "smoothness" is defined as $\frac{b^{1.5}}{v}$, and in our previous datasets [1], the smoothness of the lungs of 201 CT scans lay between $[17.11, 23.03]$. If the smoothness is not between $[17.11, 23.03]$, we mark the CT data quality as "risky". Otherwise, we say that the CT data is of good quality.

Step 4): For "risky" CT scan (about 5%), slice by slice check the data, to see whether the failure of the lung segmentation is caused by our model or caused by the bad quality of the CT.

QC for the ground truth of the heart, airways and blood vessels: we compare their volumes to the lungs and visualize the ground truth annotations in 2D and 3D to check whether they are correct. We use the following steps to analyze the qualities of ground truth annotations.

Step 1): get 3D visualizations of the annotations as well as the lung mask by STL. Check whether the ground truths are in natural shapes and positions. For airways and blood vessels, they should look like trees with affine self-similarities. The heart should be placed in the bottom middle and close to the lungs. The root of the blood vessel mask should be located inside the heart. The COVID-19 lesions should mostly appear in the lower respiratory regions where the airway and blood vessel diameters are small. If there exist problems in the relative locations or morphologies, we will discard the ground truth annotation (about 5%), otherwise move to step 2).

Step 2): get 2D visualizations of the annotations. For each ground truth annotation, we get the mass center of the mask located at $(\mathrm{x}_m, \mathrm{y}_m, \mathrm{z}_m)$. Merge the CT image with the annotation from the x-y plane at z-axis at z_m. Discard the annotation (about 5%) if it is not reasonable, otherwise to step 3).

Step 3): check the volume ratios compared to the lungs, as the volume ratios vary in small intervals. "Airways ratio" equals the volume of airways divided by the volume of the lung. Similar logic holds for the "heart ratio" and "blood vessel ratio". If the ratio is in the reasonable interval (presented in Supplementary Table 1), we say that the annotation is of good quality, otherwise we merge the CT and the annotation, then slice-by-slice check the annotation (about 2%).

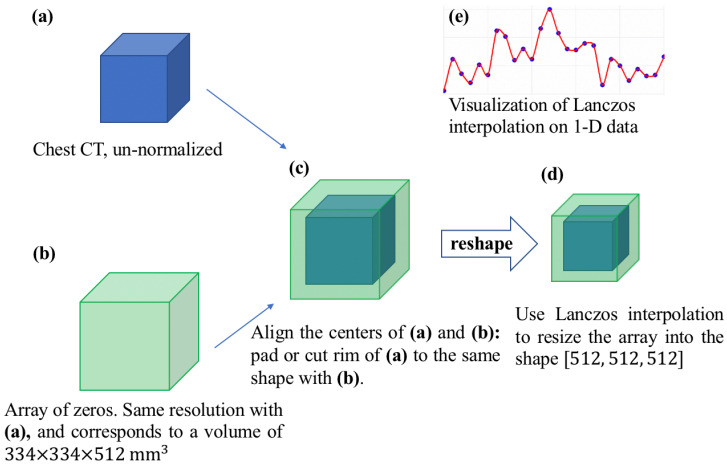
Supplementary Table 1 Reasonable volume ratios of lung tissues to the lungs. The ratio interval is formed by the [minimum ratio, maximum ratio] on a high-quality dataset containing 86 CT scans. This dataset contains CT scans from three centers and the annotations were slice-by-slice checked.

	Airways ratio	Blood vessel ratio	Heart ratio
Reasonable ratio interval	$[0.0046, 0.0246]$	$[0.0431, 0.1081]$	$[0.1040, 0.2603]$
The average of ratio	0.010796	0.064115	0.167070
The standard deviation of ratio	0.003572	0.017156	0.048748

1.1.2 Spatial normalization

During the spatial normalization, we simultaneously normalize the voxel resolution and the data dimension. Each voxel of the standard space corresponds to a volume of $\frac{334}{512} \times \frac{334}{512} \times 1.00\ \text{mm}^3$ and the standard space $\mathcal{S} \in \mathbb{R}^{512\times512\times512}$, which means $\mathcal{S}$ represents a volume of $334 \times 334 \times 512\ \text{mm}^3$. $\mathcal{S}$ has the resolution of high resolution CT (this means spatial normalization will not lose information in most cases), and $\mathcal{S}$ is big enough to completely accommodate almost all human lungs.

There are three steps to spatially normalize a chest CT scan. First, we establish a zero array representing a volume of $334 \times 334 \times 512\ \text{mm}^3$, which has the same spatial resolution with the CT scan to be normalized. For example, the spatial resolution of a CT is $1 \times 1 \times 2\ \text{mm}^3$, then the zero array should have shape $334 \times 334 \times 256$. Second, we align the center of the zero array with the CT scan. Finally, we use Lanczos interpolation to rescale the padded array into $\mathbb{R}^{512\times512\times512}$.



Supplementary Fig. 1 Illustration of spatial normalization. (a) shows the original un-normalized CT data. (b) is the array of zeros and has the same resolution with (a). (d) presents the tensor after the spatial normalization, which contains the translation ((a)&(b)→(c)) and resizing ((c)→(d)). (e) gives an illustration of the Lanczos interpolation, which is the reshape algorithm.

1.1.3 Signal normalization

Hounsfield Units (HU) is used to measure the CT signal. However, HU uses water and air to linearly normalize the CT signals, and it is suboptimal for observing pulmonary parenchyma: in our dataset, the CT signals for healthy pulmonary parenchyma vary from -400 HU to -700 HU.

In practice, different CT scanners may have default lung windows, and using the default lung window of each scanner can help alleviate the scanner biases. The lung window has two quantities: window level (WL) and window width (WW), and the signal normalization voxel-wisely transforms the original CT value $I_{original}$ of the spatial normalized $\mathcal{S}$ according to:

$$I_{normalized} = \frac{I_{original} - \text{WL}}{\text{WW}}, \tag{1}$$

In other words, the signal normalization linear changes the data based on the default lung window of each scanner, which can help alleviate the scanner signal biases.

1.1.4 CT scans with contrast agents

A small number of CT scans (148 CT scans) were collected after the injection of the contrast agents. The CT scans with contrast agents are usually from patients with pulmonary embolism, and these contrast agents are to identify arteries and veins. But CT scans in order to observe the pulmonary parenchyma do not need contrast agents. Thus, the CT scans with contrast agents contain systematic biases from CT scans that DLPE aimed to enhance, e.g., CT scans for pneumonia patients.

We use CT scans with contrast agents as a special test set to evaluate the robustness of DLPE method. If DLPE methods can run normally on these biased datasets, it implies that our methods may be suitable for even wider scenarios.

1.2 The 2.5D segmentation algorithm

The 2.5D segmentation algorithm (or "the 2.5D model") is an accurate, fast, and robust framework for 3D segmentation tasks. Based on the 2.5D model, we achieve human-level segmentations for airways, lungs, heart, blood vessels and various COVID-19 lesions. For the segmentations of airways and blood vessels, we customize the 2.5D model with special loss functions and training protocol, which are described in details in Supplementary Section 1.3 and Supplementary Section 1.4.

We train different 2.5D models for segmenting different tissues, because different human tissues vary greatly in the morphologies and usually need special loss functions and training protocols.

1.2.1 Intuitions of the 2.5D model

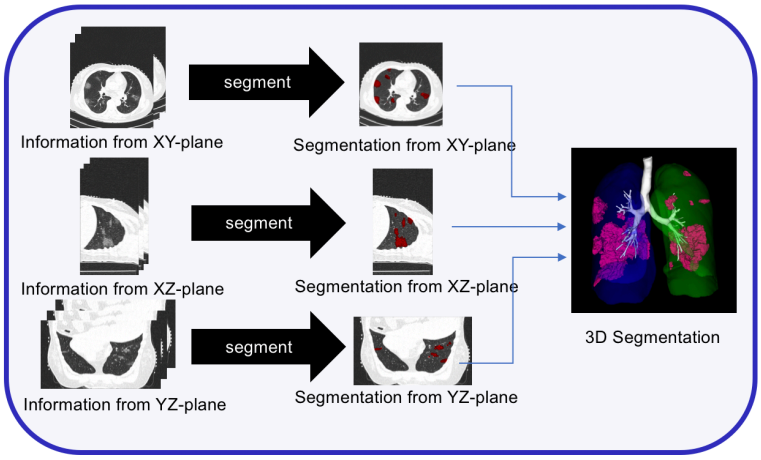
Nearly all human tissues can be identified with several 2D images. For example, when segmenting airways from a CT slice, experienced radiologists only need information from the previous slice, current slice and posterior slice. And most lesions and organs can be identified by a single CT slice, like COVID-19 lesions, pulmonary nodules, etc. Thus, when segmenting on a 2D CT slice, it is not necessary to input all the 3D data, instead, the 2D slice and its adjacent slices contain enough information, from which experienced radiologists annotated the ground truths.

The idea of the 2.5D model is to simplify the 3D segmentation task into 2D segmentation tasks from different views, and then fuses these 2D segmentations into the final 3D segmentation. It is true that simplifying the 3D task into 2D may lose some information, but the information loss should be neglectable: because the 3D ground truth is formed by stacking 2D ground truths, and the fusion of 2D segmentations from different views utilizes some 3D information.

1.2.2 The architecture of the 2.5D model

The 2.5D model contains three 2D segmentation models, and these three models are responsible for the 2D segmentations from x-y (transverse), y-z (coronal) and x-z (sagittal) planes, respectively. In this study, we use the 2D U-net for 2D segmentations, which performed best in our previous studies for both lung segmentations and COVID-19 lesion segmentations [1].

The inputs of the 2D models with shape $\mathbb{R}^{(m+n)\times 512\times 512}$, which is formed by stacking m number of adjacent 2D CT slices and n number of guidance channels (see Supplementary Section 1.4). Each 2D model outputs the segmentation probability mask in $\mathbb{R}^{512\times 512}$.

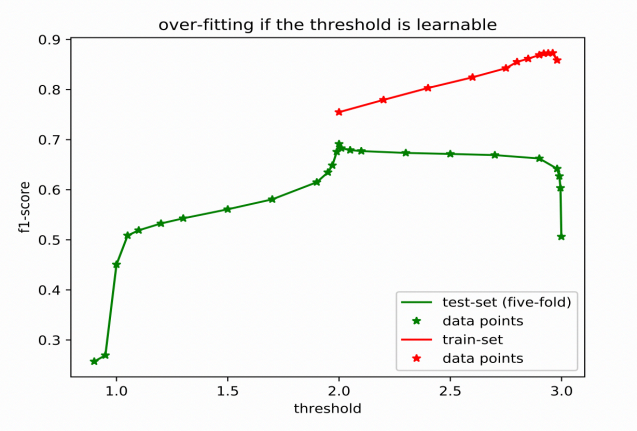


Supplementary Fig. 2. The architecture of the 2.5D segmentation algorithm. The 2.5D model contains three 2D U-net models, for x-y (transverse), y-z (coronal) and x-z (sagittal) planes, respectively. The input of each 2.5D model is several adjacent CT slices and 0 to 2 number of guidance channels (e.g., high recall mask, high precision mask). The output of each 2.5D model is the segmentation probability mask in $\mathbb{R}^{512\times512}$. We stack the predictions from different views and merge them together to get the final 3D segmentation. Because the semantics we want to segment vary greatly in the morphology, we train separate 2.5D models for segmenting lungs, heart, airways, blood vessels, COVID-19 lesions visible under the lung window and COVID-19 lesions after DLPE enhanced. And for different semantics, we customize hyper-parameters, loss functions and the training protocols.

Here we give the mathematical formulation of the 2.5D model. The inputs of the 2.5D models is the normalized machine-agnostic tensor with shape $\mathbb{R}^{512\times512\times512}$. The 2.5D model contains three 2D U-Nets, which are denoted as f_{xy} for segmenting x-y planes, f_{yz} for segmenting y-z planes, and f_{xz} for segmenting x-z planes. The input for f_{xy} is $\mathcal{P}_{\mathrm{xy}}$, the input for f_{yz} is $\mathcal{P}_{\mathrm{yz}}$, and the input for f_{xz} is $\mathcal{P}_{\mathrm{xz}}$. $\mathcal{P}_{\mathrm{xy}}$, $\mathcal{P}_{\mathrm{yz}}$ and $\mathcal{P}_{\mathrm{xz}}$ have the same shape and dimensions, which is $\mathbb{R}^{(\mathrm{m+n})\times512\times512}$, which is formed by stacking m number of adjacent CT slices and n numbers of guidance channels. All 2D models are binary prediction models, and the outputs of the 2D models are the probability map in shape $\mathbb{R}^{512\times512}$, which indicates the probabilities of the pixels to be positive semantic. By stacking the probability maps, each 2D model outputs a 3D probability mask shaped $\mathbb{R}^{512\times512\times512}$. We denote the probability mask generated by f_{xy} as $\hat{\mathrm{p}}_{\mathrm{xy}}$, and similarly we have $\hat{\mathrm{p}}_{\mathrm{yz}}$ and $\hat{\mathrm{p}}_{\mathrm{xz}}$. We then use a combination function g to fuse these three probability masks into the final binary mask. The final 3D binary mask can be presented as:

$$\hat{\mathrm{p}} = g(\hat{\mathrm{p}}_{\mathrm{xy}}, \hat{\mathrm{p}}_{\mathrm{yz}}, \hat{\mathrm{p}}_{\mathrm{xz}}). \tag{2}$$

In this study, we have three different types of combination function g. The first type is to get the final binary segmentation mask, and in this case g is defined as $\hat{\mathrm{p}}_{\mathrm{xy}} + \hat{\mathrm{p}}_{\mathrm{yz}} + \hat{\mathrm{p}}_{\mathrm{xz}} > 2$. This decision is based on our previous study [1], in which we tried many g to merge probability masks into binary prediction, including learnable weights, using adjacent probabilities to determine the semantic of the central voxel, multiplications then take the threshold, etc. We found that simply summing up and taking the threshold performed best, and learnable weights are likely to cause overfitting (Supplementary Fig. 3). The second type is to get the high recall mask, and the third type is to get the high precision mask. The second and the third types will be discussed in details in Algorithm 4 in the Supplementary Section 1.4.



Supplementary Fig. 3. In general, we find that a learnable $\boldsymbol{g}$ is likely to cause strong overfitting, while simply summing up and taking the threshold can get a robust and satisfactory segmentation. In the above figure, $\boldsymbol{g}$ is defined as $\widehat{\mathbf{p}}_{\mathbf{xy}} + \widehat{\mathbf{p}}_{\mathbf{yz}} + \widehat{\mathbf{p}}_{\mathbf{xz}} > \mathbf{threshold}$, and the figure shows how the F1-score (equals to dice in binary segmentation) of COVID-19 lesions (visible under the lung window) changes with the threshold. The best threshold on the training set is 2.96, while the best threshold for the test set is 2.00. Thus, if we let the threshold be learnable, it is likely to result in overfitting. What is interesting is that the best threshold exactly equals 2, and changing the threshold from 2 to 2.8 changes less of the F1-score. We observe very similar behaviors on the lung segmentations and the heart segmentations. Thus, we use $\widehat{\mathbf{p}}_{\mathbf{xy}} + \widehat{\mathbf{p}}_{\mathbf{yz}} + \widehat{\mathbf{p}}_{\mathbf{xz}} > \mathbf{2}$ for the final binary segmentation.

1.2.3 Performance evaluation metrics

The 2.5D models are segmentation models, and the dice score, recall, precision are commonly used metrics for the model evaluations. These metrics are defined as follows.

The dice score, or dice, or dice similarity coefficient, is defined as:

$$\text{Dice}: = \frac{2|Y \cap Y'|}{|Y| + |Y'|}, \tag{3}$$

Here Y and Y' are two sets, i.e., Y is the ground truth binary mask for the positive semantic and Y' is the predicted binary mask for the positive semantic. In binary classification, dice equals to the harmonic average of precision and recall (the F1-score):

$$\text{Dice} = \frac{2}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} \tag{4}$$

All dice scores in this study are calculated at a scan-level.

In order to analyze the lesions segmented by the 2.5D models, we quantify the radiomics and use the Pearson Correlation Coefficient (PCC), the mean absolute error (MAE) and the root mean square error (RMSE).

$$\text{PCC} = \frac{cov(Z, Z')}{\sigma_Z \sigma_{Z'}}, \tag{5}$$

where Z is a variable, like a radiomic, and Z' is another variable, like the SGRQ score. $cov(Z, Z')$ gives the covariance between Z and Z'. σ_Z, $\sigma_{Z'}$ are the standard deviation of Z and Z', respectively.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - y_i'|, \tag{6}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - y_i')^2}, \tag{7}$$

where y_i is the target feature for the i-th patient, like the PaO2/FiO2 ratio, and y_i' is the predicted value.

In addition, we present a new metric focusing on the segmentations of tiny airways and blood vessels (region discovery dice, see Algorithm 3).

1.2.4 Important setups for the 2.5D models

In our previous study [1], we found that the 2.5D model is not sensitive to the learning rate, batch size and the optimizer when segmenting COVID-19 lesions and the lungs. Thus, for all 2.5D models in this study, we fix the batch size to 64, learning rate to 10^{-4} and use the Adam optimizer. However, our previous study [1] found that the 2.5D models are usually sensitive to the following setups, which are shown in the Supplementary Table 2:

Supplementary Table 2 The setups that have significant influences on the model performance. The first row is the setups for the 2.5D model, and the other rows give the candidate setups. Columns from left to right: the loss function for the 2D U-net (feature enhanced loss only applicable for segmentations of airways and blood vessels); the adjacent CT slices, 0 means that the CT slice needs to be segmented, $-k$ means the previous k CT slice and $+k$ means the posterior k CT slice; the training protocol is explained in Supplementary Section 2.5; the "initial features" means the number of convolutional kernels for the first layer of the U-net. The highlighted candidates are the defaulted setups, which were the best setups when segmenting the COVID-19 lesions in our previous study [1].

Loss function	Adjacent slices	Training protocol	Initial features
Dice loss	$(0,)$	Fluctuate	8
Cross-entropy loss	$(-1, 0, +1)$	Direct converge	16
Feature enhanced loss	$(-5, -2, 0, +2, +5)$		32
	$(-8, -5, -2, 0, +2, +5, +8)$		

1.2.5 Training protocols for the 2.5D model

For some tasks, the 2.5D model has a high risk of sticking to the local minimum. Thus, we present the fluctuate training protocol in order to help the model converge to better solutions. Fluctuate training protocol changes the global class-balance weights of the loss function, which changes the relative penalties between false predictions of classes, thus drags the model out of the local minimum. By comparison, the "direct converge" training protocol follows the traditional training procedures: train the model until the loss is converged or the model starts to over-fitting. The fluctuating training protocol needs 3-5 times longer training time, thus, if there is no significant improvement, we will use the direct converge training protocol.

Algorithm 1: Direct converge training protocol

Inputs: initial model, dataset, loss function

Output: trained model

```
training set, validation set, test set = dataset_manager(dataset)
converged = False
while not converged:
    model_trainer(initial model)          # Here train the initial model for an epoch
    if loss on validation set start to increase:
        converged = True
    if loss on training set is not decrease:
        converged = True
return initial model
```

Algorithm 2: Fluctuate training protocol

Inputs: initial model, dataset, loss function

Output: trained model

```
training set, validation set, test set = dataset_manager(dataset)
sub-optimal model = Direct converge training protocol(initial model, dataset, loss function)
while recall < 2 precision:                    # Calculated on the validation set
    false negative penalty = 1.05 false negative penalty    # Change the loss function to give more penalties
    model_trainer(sub-optimal model)           # Here train the model for an epoch
while precision < 2 recall:                    # Calculated on the validation set
```

7. false positive penalty = **1.05** false positive penalty # *Change the loss function to give more penalties*
8. model_trainer(sub-optimal model) # *Here train the model for an epoch*
9. reset the loss function to the original one # *The original is class-balanced*
10. trained model = Direct converge training protocol(sub-optimal model, dataset, loss function)
11. **return** trained model

1.2.6 Workflow for seeking optimal setups for the 2.5D models

There are six 3D segmentation tasks in our study: the segmentations for the heart, lungs, airways, blood vessels, COVID-19 lesions visible under the lung window, and COVID-19 lesions after DLPE enhanced. All of them use the 2.5D segmentation algorithm, but they have different optimal hyper-parameters, and apply different loss-functions and training protocols.

In order to seek optimal setups for these 2.5D models, we randomly select 200 CT scans for the segmentation problems of lungs, the heart, airways, blood vessels and COVID-19 lesions visible under the lung window. And use all the 173 CT scans from the COVID-19 survivors.

For the lesions discovered by DLPE method, we use a human-in-the-loop annotation approach. Initially, we do not have the pixel-level ground truth and only have the region of interest level ground truth. This makes the dice score inappropriate to evaluate the performance of the model. Thus, we use the radiologists' rating to evaluate the performance of the model.

Supplementary Table 3 The datasets and the performance metrics for seeking the best loss function, training protocol and hyper-parameters. For sub-visual lesions detected by DLPE method, we use a human-in-the-loop annotation approach. Thus, initially there is no pixel-level ground truth and radiologists directly rate models with different setups.

Target semantic	Performance measurement	Dataset size	Cross-validation
Lungs	Dice	200	5-fold
Heart	Dice	200	5-fold
Airways	Dice	200	5-fold
Blood vessels	Dice	200	5-fold
COVID-19 Lesions (lung window)	Dice	200	5-fold
COVID-19 Lesions (DLPE enhanced)	Radiologists' rating	173	10-fold

We use a greedy approach to approximate the optimal setups. There are four setups to be determined, i.e., the loss function, adjacent slices, training protocol and initial features. The greedy approach splits the problem into four sub-problems: for each of the four setups, finding the best setup among its candidates while the other three are fixed to the default value (highlighted in Supplementary Table 1). Combining the output of the four sub-problems, we get the final setups. Thus, for each 2.5D model, we need to evaluate at most 12 setups.

In general, refining these setups do improve the dice performance compared to the default setup for the 2.5D model, and the improvement is usually around 0.1 of dice score. The 2.5D method is quite stable and robust, and the workflow is able to find good setups for the 2.5D models to get satisfactory segmentations.

1.2.7 The optimal setups for the 2.5D models

We train eight 2.5D models for the six 3D segmentation tasks. All the eight 2.5D models use the batch size of 64, learning rate of 10^{-4} and use the Adam optimizer. The setups and other details for the eight 2.5D models are presented in Supplementary Table 4.

Supplementary Table 4 The setups for the eight 2.5D models in our study. The models are named by their segmentation target. In the "Loss function" column, "Dice" means that the 2D U-net uses the Dice loss, and the same logic holds for other loss function names. Guidance channel is not a hyper-parameter and is only applicable for the second-stage model of the two-stage segmentation protocol, thus, other 2.5D models are "NA (not applicable)".

Model name	Loss function	Adjacent slices	Guidance channel	Training protocol	Initial features
Lung	Dice	$(-1, 0, +1)$	NA	Direct converge	16
Heart	Dice	$(-1, 0, +1)$	NA	Direct converge	16

Airway-first-stage	Feature enhanced	$(-5,-2,0,+2,+5)$	NA	Fluctuate	32
Airway-second-stage	Feature enhanced	$(-1,0,+1)$	High recall, high precision	Fluctuate	16
Blood vessel-first-stage	Feature enhanced	$(-1,0,+1)$	NA	Fluctuate	16
Blood vessel-second-stage	Feature enhanced	$(-1,0,+1)$	High recall, high precision	Fluctuate	16
COVID-19-visible	Cross-entropy	$(-5,-2,0,+2,+5)$	NA	Direct converge	32
COVID-19-DLPE	Cross-entropy	$(-1,0,+1)$	NA	Direct converge	16

1.2.8 The visual interpretation methods for the 2.5D models

The visual interpretation helps the understanding of the models and guides the algorithm design. Grad-Cam [21] and its variants are widely used techniques for the visualizations of convolutional neural networks (CNN). The intuition of Grad-Cam is that the locations in the feature maps reflect the locations of the information sources: the pixel at the center of the feature map is influenced more by center parts of the inputs, while pixels at the upper parts of the feature maps are influenced more by the upper parts of the inputs, etc. Using this property, Grad-Cam is able to visualize the discriminative regions for the outputs.

Different layers reflect different features, and it is very flexible to get the visual interpretations based on Grad-Cam, especially for segmentation models. Vinogradova et al [22] applied Grad-Cam on the bottleneck layers of the 2D U-net for visualizing discriminative regions. On the other hand, applying Grad-Cam on the last convolutional layer of the 2D U-net reflects the feature importance map. In the 2D U-net, the last convolutional feature maps have the same width and height with the outputs. Each pixel of the last convolutional feature maps has already gathered features for the final output, and the U-net only needs to apply a simple 1×1 convolution on the last convolutional feature maps to output the final predictions. Thus, if a pixel in the last feature maps strongly affects the final predictions, it implies that this pixel gathers important features. Similar ideas have been used by Kristoffer et al [43] for the feature importance map.

Here we present the mathematical definitions for the discriminative regions and the feature importance map. The 2.5D model merges 2D predictions from three views, i.e., the x-y plane, the y-z plane and the x-z plane. For simplicity, the visual interpretation focuses on the predictions from the x-y planes. The 2D segmentation model for x-y planes outputs the semantic map with shape $\mathbb{R}^{2\times512\times512}$, which is before softmax. The first channel is for the negative semantic, denoted as P^n; the second channel is for the positive semantic (like the COVID-19 lesions), denoted as P^p. After softmax, we have the probability map for positive semantic, which with is shaped 512×512 and is denoted as P. The stack of feature maps for a convolutional layer is denoted as A. The number of feature maps for A equals the number of the convolutional kernels, and the k-th feature map is denoted as A^k.

Segmentation is a dense prediction task, thus the 2D segmentation task is split into 512×512 numbers of smaller classification tasks. For each classification task, we can use the Grad-Cam method to calculate discriminative regions and the feature importance map. Similar with [21], the $Pixel\_Heat\_Map$ for the pixel at row i column j is defined as follows:

$$Pixel\_Heat\_Map \triangleq^{\text{def}} ReLU(\sum_k \alpha_{ij}^k A^k) \tag{8}$$

Where $ReLU(\cdot)$ means replace negative values in the input tensor with zero, and α_{ij}^k is a real number given by:

$$\alpha_{ij}^k = \text{sum}(\frac{\partial(P_{ij}^p - P_{ij}^n)}{\partial A^k}) \tag{9}$$

Here the $\text{sum}(\cdot)$ function means summing up all elements of the input tensor.

Because we aim to visualize how the model predicts the positive semantic, we define the final heatmap to be the weighted superposition of all $Pixel\_Heat\_Map$ according to the probability P_{ij}. The final heatmap is given as follows:

$$Final\_Heat\_Map \triangleq^{\text{def}} ReLU(\sum_{ij} P_{ij} \sum_k \alpha_{ij}^k A^k) \tag{10}$$

Note that:

$$\sum_{ij} P_{ij} \sum_{k} \alpha_{ij}^{k} A^{k} = \sum_{k} \sum_{ij} P_{ij} \alpha_{ij}^{k} A^{k} = \sum_{k} \sum_{ij} \frac{e^{P_{ij}^{t} - P_{ij}^{n}}}{1 + e^{P_{ij}^{t} - P_{ij}^{n}}} \alpha_{ij}^{k} A^{k} \tag{11}$$

And we have:

$$\sum_{ij} \frac{e^{P_{ij}^{t} - P_{ij}^{n}}}{1 + e^{P_{ij}^{t} - P_{ij}^{n}}} \alpha_{ij}^{k} = \text{sum}\left(\frac{\partial \text{sum}(\ln(1 + \exp(P^{t} - P^{n})))}{\partial A^{k}}\right) \stackrel{\text{def}}{=} \beta^{k} \tag{12}$$

Here the $\exp(\cdot)$ function means getting the exponent for every element of the input tensor. With equation (12), the final heatmap can be easily calculated by:

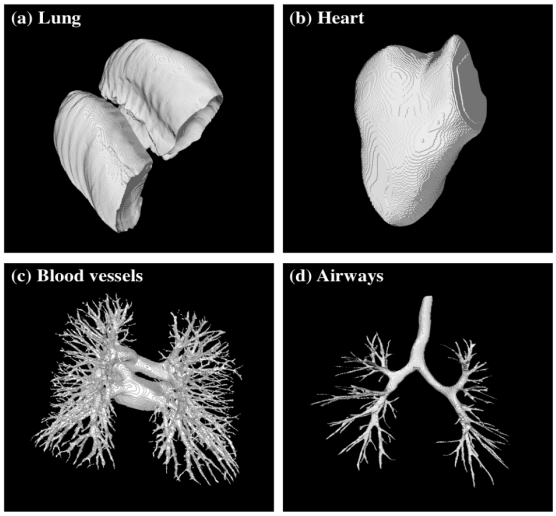
$$Final\_Heat\_Map = ReLU\left(\sum_{k} \beta^{k} A^{k}\right) \tag{13}$$

As explained previously, if A is the feature maps of the bottleneck layer, then the heatmap shows the discriminative regions; if A is the stack of feature maps of the last convolutional layer, then the heatmap gives the feature importance map. In Supplementary Section 1.4.5, we will use these visual interpretation methods to analyze our models.

1.2.9 A simple method to evaluate the segmentations of 2.5D models

The 2.5D model is not fed any 3D structure information during training and the fusion of different views. During the training and the fusing of different views, the 2.5D model only receives information from 2D cross-sections, and has no information about the 3D long-range correlations of the human tissues. For example, during training, the model receives no information about the 3D visualization of the lung mask, which has 4-7 parallel grooves caused by the ribs. Similarly, the model does not know that the 3D visualizations of the blood vessels and airways are tree-like structures with affine self-similarity. The model also does not know that the heart, airways and blood vessels only have one 3D connective component.

Thus, if the 3D segmentation by the 2.5D model shows natural 3D structures, it is likely that the segmentation is of good quality. Supplementary Fig. 4 gives illustrations about the "natural structure" for the lungs, airways, blood vessels and the heart.



Supplementary Fig. 4 The human-comparable segmentation results of the 2.5D models on a normal Chest. (a) The lung mask should see clear parallel grooves (caused by the rib) on the surface. (b) The surface of the heart mask should be relatively smooth. (c, d) Blood vessels and airways are tubes with tree-like structures.

1.2.10 Region discovery dice to evaluate the robustness of the 2.5D models

In our previous study [1], the 2.5D model was proved to have excellent robustness as it can accurately segment COVID-19 lesions and lung masks for chest CT scans from different countries. In this study, we focus on the robustness of the airways and the blood vessels.

We use the region discovery dice (RDD) to measure the robustness. Two reasons lead to the RDD. First is that the dice score cannot quality the segmentation performance for tiny structures. As discussed in Supplementary Section 1.3.3, if a model neglects the smallest 50% airway cross-sections, it can still reach a dice score of 0.9659 in our datasets.

The second reason is that the annotation styles for the blood vessels on CT scans with contrast agents are quite different. As explained earlier, we will evaluate the robustness of the 2.5D model on the CT scans with contrast agents. We observe that the annotations of the blood vessels near the heart are quite conservative and rough. This is because for the CT scans with contrast agents, radiologists aim to separate arteries and veins, and near or inside the heart the arteries and veins are very close to each other. Thus, the ground truths for these CT scans near the heart may overlook ambiguous regions even if they are clear blood vessels, and the traditional dice score (equation (3)) is suboptimal here.

The "region discovery dice (RDD)" is very similar with the dice score: dice score treat all voxels as equal while RDD treat all connected components as equal. In the dice score, we calculate each voxel to see whether the voxel is predicted correctly. In RDD, we calculate each connective component in x-y planes (as radiologists annotated on x-y planes), and see whether the prediction and the ground truth have overlap. The vale of RDD is between 0 and 1. When dice score is 0, RDD is always 0, and when dice score is 1, RDD is always 1. RDD is defined as follows:

Algorithm 3: Region discovery dice

Inputs: predicted 3D binary mask ($\mathbf{Y}'$), ground truth 3D binary mask ($\mathbf{Y}$)
 # $\boldsymbol{Y}'$ and $\boldsymbol{Y}$ are binary arrays both with shape [512, 512, 512]
Output: Region discovery dice

```
num_connected_regions_Y = 0                                  # Calculated on the x-y planes
num_connected_regions_Y' = 0                                 # Calculated on the x-y planes
# num_connected_regions_Y + num_connected_regions_Y' is similar with |Y| + |Y'| in equation (3)
num_overlap_region_count = 0                                 # Calculated on the x-y planes
# num_overlap_region_count is similar with 2|Y ∩ Y'| in equation (3)
for z in range(0, 512):
    list_ connected_regions_Y = get_connected_component(Y[:, :, z])
    list_ connected_regions_Y' = get_connected_component(Y'[:, :, z])
    num_connected_regions_Y += len(list_ connected_regions_Y)
    num_connected_regions_Y' += len(list_ connected_regions_Y')
    for connected_region in list_ connected_regions_Y:
        for connected_region' in list_ connected_regions_Y':
            if connected_region ∩ connected_regions' is not None:
                num_overlap_region_count += 1
                break
    for connected_region' in list_ connected_regions_Y':
        for connected_region in list_ connected_regions_Y:
            if connected_region' ∩ connected_regions is not None:
                num_overlap_region_count += 1
                break
return num_overlap_region_count / (num_connected_regions_Y + num_connected_regions_Y')
# Same logic with equation (3)
```

1.3 The feature-enhanced loss

In the main text, we analyzed the relationship between the area A (the area of the connected component on x-y planes, which is an integer) and the frequency f, and found that the f-A relationship follows the power law function, which is $f \sim A^{-\gamma}$:

For blood vessels, we analyzed 1594446 regions and find $\gamma = 1.92$:

$$\ln(f) = -1.92\ln(A) + 17.1, r = 0.9944 \tag{14}$$

For airways, we analyzed 420667 regions and find $\gamma = 1.75$:

$$\ln(f) = -1.75\ln(A) + 15.0, r = 0.9961 \tag{15}$$

For airways, the average region area is 73.70 pixels, while the median region area is 21 pixels. For blood vessels, the average region area is 81.06 pixels, while the median region area is 19 pixels. The significant difference in the average and the median implies that most regions are small regions but the total volume of small regions are neglectable. This is caused by the properties of the power-law function.

1.3.1 Small airways and blood vessels are neglectable in the volume

The total volumes for airways and blood vessels within cross-section area range $[A_{small}, A_{large}]$ is given by:

$$\boldsymbol{voxel\_volume} \sum_{\boldsymbol{A_{small}}}^{\boldsymbol{A_{large}}} \boldsymbol{f(A)} \tag{16}$$

Here the $voxel\_volume$ equals to $\frac{334}{512} \times \frac{334}{512} \times 1.00\ \text{mm}^3$, which is the normalized spatial resolution. For simplicity, we change the summation Σ into integral, as the resolution is quite high and the interval of Σ is very small. $f(A)$ is the count of how many regions have A pixels. We define "small region" as the regions with the area less than a.

In the power-law relationship which is $\ln(f) = -\gamma\ln(A) + c$ or $f = c_0 A^{-\gamma}$, $\gamma = 2$ is a watershed:

$$\lim_{\varepsilon \to 0^+} \int_{\varepsilon}^{2\varepsilon} f dA = \begin{cases} 0 & \text{if } 1 < \gamma < 2 \\ c_0 \ln(2) & \text{if } \gamma = 2 \\ \infty & \text{if } 2 < \gamma \end{cases} \tag{17}$$

And for $1 < \gamma < 2$ we have:

$$\int_0^a f dA = \frac{c_0 a^{2-\gamma}}{2-\gamma} \text{ while } \int_a^{\infty} f dA = \infty \tag{18}$$

Note that in the f-A relationship, $\gamma = 1.92$ for blood vessels and $\gamma = 1.75$ for airways, which are smaller than 2. Then, equation (17) shows that, the total volumes for small airways and small blood vessels are neglectable compared to the total volumes of the large airways and large blood vessels.

1.3.2 Most regions are small

Despite that small regions constitute a neglectable volume, most regions are small regions. For $1 < \gamma$, we have:

$$\lim_{\varepsilon \to 0^+} \int_{\varepsilon}^{a} f dA = \infty \text{ while } \int_a^{\infty} f dA = \frac{a^{1-\gamma}}{\gamma - 1} \tag{19}$$

Equation (19) means that the number of small regions is much larger than that of big regions when ε is small.

On our datasets, the volume summation of the smallest 50% (regions with area that is less or equals to the median area) is only 6.60% of the total volume for airways, and is only 6.11% of the total volume for blood vessels.

1.3.3 Dice loss and cross-entropy loss are sub-optimal for blood vessels and airways

The above analysis indicates that the traditional voxel-wise loss and performance metrics like dice score may neglect small airways and blood vessels. Consider two segmentation models, model A and model B, where model B is the perfect model while model A is perfect for bigger airways but cannot detect any of the small airways with cross-section area that is less or equals to 21 pixels (this means the final segmentation of model A will neglect 50% of small regions when viewing on the x-y plane).

A good loss function should give Model B much less loss than model A. However, the cross-entropy loss and the dice loss for model A and model B are nearly the same.

Take the dice loss as an example. Let G be the ground truth mask, and P be the prediction mask, then the dice loss is defined as:

$$Dice\_Loss = 1 - \frac{2|\mathrm{G} \cap \mathrm{P}|}{|\mathrm{G}| + |\mathrm{P}|} \tag{20}$$

The dice loss ranges from $[0, 1]$. For the model B, which is the perfect model, the dice loss is 0. The model A neglects all small airways with cross-section less than 21 pixels, and these small airways constitute 6.60% of the total volume for airways. Thus, if the precision for model A is perfect, the dice loss for the model A is:

$$Dice\_Loss(Model\ A) = 1 - \frac{2 \times 0.934}{1 + 0.934} = 0.0341 \tag{21}$$

Dice loss of 0.0341 is extremely small, and by comparison, for airways, the dice losses between the human annotations of the same CT from different radiologists are usually around 0.05. This means that using the dice loss hampers the model to learn fine structures, as the model only needs to detect big airways or blood vessels. Similar logic holds for the cross-entropy loss.

1.3.4 Voxel-wise weight balance

We add a voxel-wise weight on the cross-entropy loss to balance the problem of neglecting small airways and small blood vessels. The weights change the penalties of the false positive and the false negative:

$$weighted\ loss = -w_{fn} \times \ln(p) \times p' - w_{fp} \ln(1 - p) \times (1 - p') \tag{22}$$

where p is the predicted probability that the voxel is positive, and p' is the ground truth probability that the voxel is positive. w_{fn} regulates the false negative penalty, while w_{fp} regulates the false positive penalty. As the ground truth is binary, we need to define a w_{fn} if the voxel is positive (p' equals to one), and define a w_{fp} if the voxel is negative ($(1 - p')$ equals to one).

1.3.5 $\overline{w}_{fn}$ to balance each branching level

We let the average w_{fn} inside a region only depend on the region area, i.e., the average w_{fn} inside a region with area A is denoted as $\overline{w}_{fn}(A)$. The idea for w_{fn} is to give equal focus for every branching level, because airways and blood vessels have self-affine similarities, which implies that each branching level contains a similar amount of information. As discussed in the main text, the cross-section area for airways and blood vessels at branching level l, roughly satisfies the relationship of $A_l = A_0 \alpha^l$, here $0 < \alpha < 1$. Thus, the area between branching level i and $i + 1$ should belong to $[A_0 \alpha^{i+1}, A_0 \alpha^i)$, and the summation of w_{fn} for all voxels of branching level i and $i + 1$ is given by:

$$\int_{A_0 \alpha^{i+1}}^{A_0 \alpha^i} f \overline{w}_{fn}(A) A \, dA \tag{23}$$

And we want equation (23) to be a constant for all branching level i. A simple solution is to let:

$$\overline{w}_{fn}(A) = c_1 A^{\gamma - 2} \tag{24}$$

Where c_1 is a constant. Note that $f = c_0 A^{-\gamma}$, and then equation (23) reduced to:

$$\int_{A_0 \alpha^{i+1}}^{A_0 \alpha^i} c_0 A^{-\gamma} c_1 A^{\gamma - 2} A \, dA = c_0 c_1 \int_{A_0 \alpha^{i+1}}^{A_0 \alpha^i} A^{-1} \, dA = -c_0 c_1 \ln(\alpha) \tag{25}$$

Thus, $\overline{w}_{fn}(A) = c_1 A^{\gamma - 2}$ makes the penalties for each branching level to be $-c_0 c_1 \ln(\alpha)$, which is a constant. The total penalty weights for positives and negatives should be class-balanced, thus, once $\overline{w}_{fn}(A)$ is determined, we can get the total summation of w_{fp}:

$$\sum_{all\ positives} w_{fn} = \sum_{all\ negatives} w_{fp} \tag{26}$$

1.3.6 w_{fn} and w_{fp} for the feature enhanced loss

$\overline{w}_{fn}$ determines the average w_{fn} for each cross-section region, we then specify the spatial distribution of w_{fn} inside each region. We have two intuitive plans: Plan A is to let each voxel in the region have the same weight, i.e., each voxel has weight $\overline{w}_{fn}$; Plan B is to let each voxel have weight $\frac{1}{2}\overline{w}_{fn}$, and equally increase the weights for boundary voxels so that the average weight becomes $\overline{w}_{fn}$.

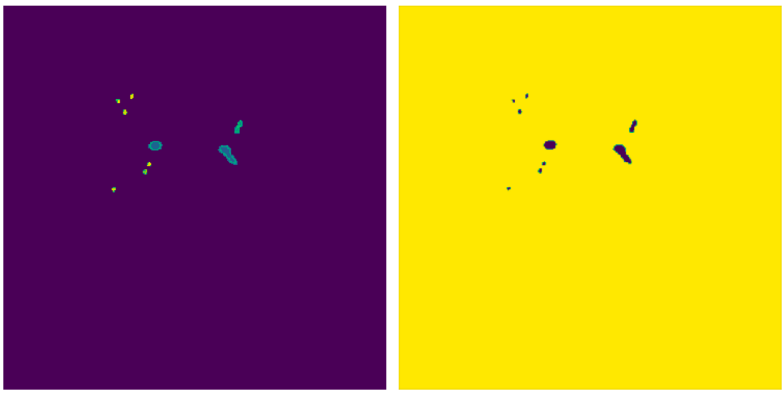
Similarly, we try two plans for the distribution of w_{fp}. Denote the average w_{fp} for all negative voxels to be $\overline{w}_{fp}$, and in this study, we set $\overline{w}_{fp}$ to 1. Plan C is to let each negative voxel has the same w_{fp}, i.e., 1; Plan D is to let each negative voxel has weight $\frac{1}{2}$, while equally increase the weights for outer surface of the positive regions so that the average weight become 1.

When using the "direct converge" training protocol and the airway segmentation datasets with 200 CT scans, we find that Plan D makes the model slip into local optimums, while using Plan B can significantly accelerates the training speed (Supplementary Table 5).

Supplementary Table 5 Evaluation of the spatial distributions of $\boldsymbol{w_{fn}}$ and $\boldsymbol{w_{fp}}$. Plan A and Plan B are for $\boldsymbol{w_{fn}}$, while Plan C and Plan D are for $\boldsymbol{w_{fp}}$. We evaluate these plans on the airway segmentation datasets with 200 CT scans. The following table shows the convergence speed and the converged training loss (the average training loss for each sample of the 2D model for x-y planes). We find that Plan B + Plan C outperforms other combinations.

	Plan A + Plan C	Plan A + Plan D	Plan B + Plan C	Plan B + Plan D
Epochs to converge	137	161	70	138
Converged training loss	748.7	3228.2	577.4	2715.7

Thus, the spatial distribution for w_{fn} follows Plan A, and the w_{fp} fixed to 1 for all negative voxels. We then have the final feature-enhanced loss, which is the equation (1) in the main text. Supplementary Fig. 5 gives direct impressions for w_{fn} and w_{fp}.



Supplementary Fig. 5 Illustrations of the weights for the feature-enhanced loss. Left shows the distribution of w_{fn}. For each region in the cross-section image, the average w_{fn} for each region follows $\overline{w}_{fn}(A) = c_1 A^{\gamma-2}$, while for the boundary voxels, we distribute more penalty weights in order to improve the convergence speed during training. The constant c_1 is to ensure the total weights for positives and negatives are class balanced. Right shows the distribution of w_{fp}. For every negative voxel, the $w_{fp} = 1$.

1.4 The two-stage segmentation protocol

1.4.1 Intuitions of the two-stage segmentation protocol

The segmentations of the airways and the blood vessels are large-scene-small-object problem. For example, the average airway volume is 41.7 cm^3, which only constitutes 0.073% the total volume of the standard embedding space ($334 \times 334 \times 512$ mm^3). For the large-scene-small-object problem, the spatial scales for the features of the background and the target vary greatly, thus the model has difficulties to simultaneously extract features for background and the target. In addition, the target is very small in the size, which means that its features may be influenced more by noise and thus difficult to be extracted by the model. This conforms to our experimental results: the 2.5D model with feature-enhanced loss achieves state-of-the-art performance, however, the segmentations for the tiny structures are not very natural: the segmented boundaries may zigzag, and are not smooth or continuous, and the dice performance for branching $level > 5$ is 0.52 for tiny airways and 0.80 for tiny blood vessels.

Recent works for the large-scene-small-object problems usually follow the idea of multiscale fusion and the coarse-to-fine approach. Multiscale fusion tries to combine the low-level and the high-level features, which is one of the most intrinsic reasons for the success of the U-net architecture: the low-level feature map will be

concatenated to the high-level feature maps. However, the 2.5D model itself does not use the coarse-to-fine approach.

The two-stage segmentation protocol is an instance of the coarse-to-fine approach. The first-stage is a coarse model, which outputs two coarse masks: a high recall mask and a high precision mask. The coarse masks narrow down the search space of the second-stage model by thousands of times. Thus, guided by these coarse masks, the second-stage model is able to output the human comparable segmentations.

1.4.2 The calculation of the high recall mask and the high precision mask

The high recall mask is gained by picking out a large number of voxels that are predicted with highest probabilities, while the high precision mask is gained by picking out a small number of voxels that are predicted with highest probabilities. The segmentations of the lungs and the heart are much easier, and we find that despite the volumes for airways and blood vessels of different people varies a lot, their relative volume ratio to the lungs and to the heart is stable. Supplementary Table 6 gives the relative ratios and their standard deviations, and the table shows that for airways, we should use the lung volume to get the high recall mask and the high precision mask, while for blood vessels we should use the heart volume.

Supplementary Table 6 The relative volume ratio for airways and blood vessels to lungs and to the heart on our datasets. The first row shows the average of the relative volume ratio for these tissues. The second row shows the standard deviations of these ratios. The third row shows the second row dividing the first row. From the third row we know that the volume ratio of airways to lungs is more stable than the ratio to the heart, while the volume ratio of blood vessels to the heart is more stable than the ratio to the lungs. Thus, for airways, we should use the lung volume to get the high recall mask and the high precision mask, while for blood vessels we should use the heart volume.

	Airways to lungs	Blood vessels to lungs	Airways to heart	Blood vessels to heart
Average volume ratio	0.01102	0.06528	0.07251	0.39433
Standard deviation	0.00354	0.01756	0.03593	0.08328
Std divide Average	0.32206	0.26905	0.49556	0.21121

In our dataset, at the scan-level, when discarded the largest 1% and the lowest 1%, the airway volume divides the lung volume varies in $[0.00453, 0.02458]$, while for blood vessels, the volume ratio to the heart varies in $[0.2748, 0.7030]$.

Thus, for airways, the high recall mask picks out 0.02458 of the lung volume, and the high precision mask picks out 0.00453 of the lung volume. For blood vessels, the high recall mask picks out 0.7030 of the heart volume, and the high precision mask picks out 0.2748 of the heart volume. Supplementary Fig. 6 gives illustrations of the high recall mask and the high precision mask.

Experiments show that the high recall masks have an average scan-level recall of 0.95 at the precision 0.56, while the high precision masks have an average scan-level precision of 0.93 at the recall 0.65. Thus the high recall mask and the high precision mask give good guidance for the second-stage model, and the second-stage model only needs to focus and refine for very small regions.

Algorithm 4: Combination functions for the high recall and the high precision masks

Inputs: rescaled CT, the first-stage model, lung segmentation binary mask (**L**), heart segmentation binary mask (H)
 # Rescaled CT, lung mask L and heart mask H are arrays in shape [512, 512, 512].
 # The lung mask L is predicted or provided by the ground truth.

Output: High recall mask, High precision mask

 # The outputs are binary arrays

1. $\widehat{\mathbf{p}}_{\mathbf{xy}}, \widehat{\mathbf{p}}_{\mathbf{yz}}, \widehat{\mathbf{p}}_{\mathbf{xz}}$ = first-stage model(rescaled CT)

   # The probability maps for the three views.

2. $\mathbf{P} = (\widehat{\mathbf{p}}_{\mathbf{xy}} + \widehat{\mathbf{p}}_{\mathbf{yz}} + \widehat{\mathbf{p}}_{\mathbf{xz}})/\mathbf{3}$

   *# **P** is the probability map combines three views*

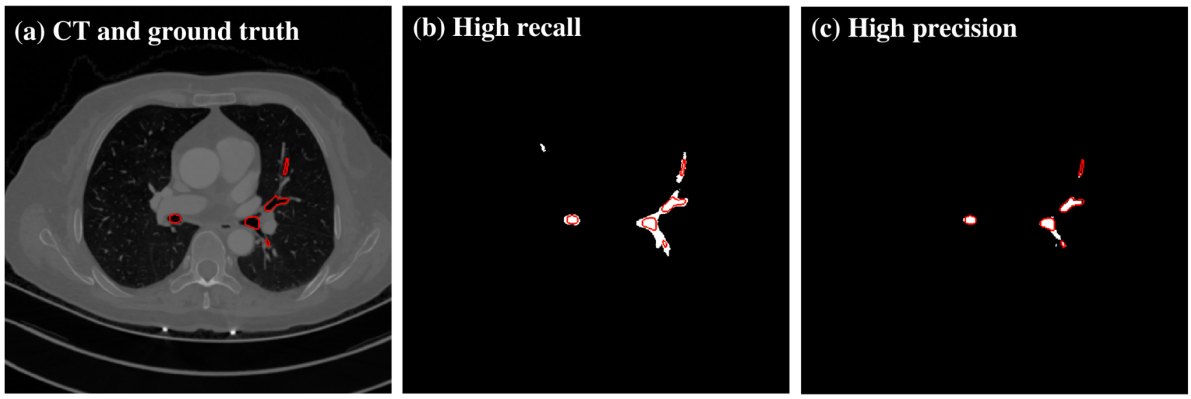
3. **if** first-stage model.target is "airways":

4. lung_voxel_count = $\mathbf{sum}(\mathbf{L})$

   # Calculate the number of voxels inside the lungs

5. voxels_for_high_recall = $\mathbf{int}(\mathbf{0.02458} \times \mathbf{lung\_voxel\_count})$

6. voxels_for_high_precision = $\mathbf{int}(\mathbf{0.00453} \times \mathbf{lung\_voxel\_count})$
7. **else:**
8. *# This means the target is "blood vessels"*
9. heart_voxel_count = $\mathbf{sum}(\mathbf{H})$
 # Calculate the number of voxels inside the heart
10. voxels_for_high_recall = $\mathbf{int}(\mathbf{0.7030} \times \mathbf{heart\_voxel\_count})$
11. voxels_for_high_precision = $\mathbf{int}(\mathbf{0.2748} \times \mathbf{heart\_voxel\_count})$
12. probability_list = flatten_then_sort($\mathbf{P}$)
 # The probability_list is from large to small
13. threshold high recall = probability_list[voxels_for_high_recall]
14. threshold high precision = probability_list[voxels_for_high_precision]
15. High recall mask = P > threshold high recall
16. High precision mask = P > threshold high precision
17. **return** High recall mask, High precision mask



Supplementary Fig. 6 Cross-section images from the x-y plane of the high recall mask and the high precision mask for the airways. (a) is the rescaled CT with the red lines showing the boundaries of the ground truth. The white regions in (b) give the high recall mask, with red lines showing the boundaries of the ground truth. The white regions in (c) give the high precision mask, with red lines showing the boundaries of the ground truth.

1.4.3 Get the final segmentation

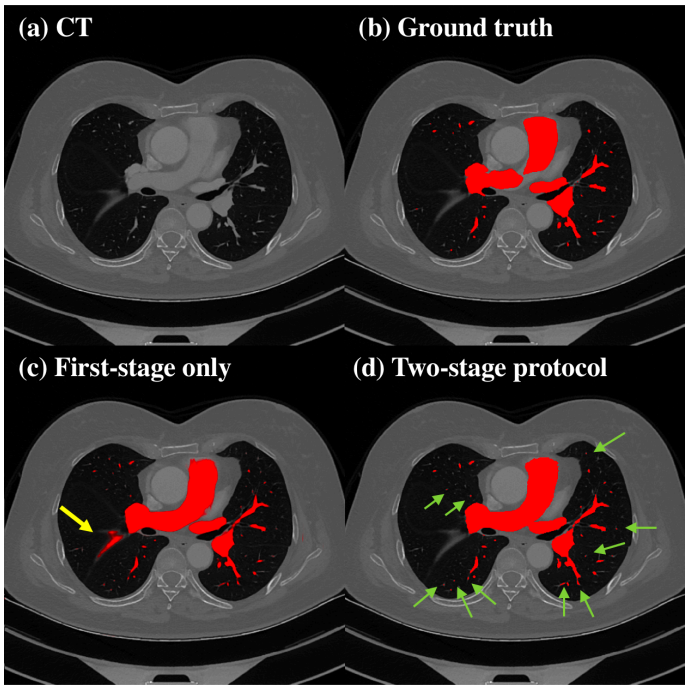
In Supplementary Section 1.2.2 we discussed the input shape for the 2D U-net in our 2.5D model. The second-stage model is also a 2.5D model, and the input shape for its 2D U-nets are in shape $\mathbb{R}^{(3+2)\times 512\times 512}$, as the input is the stack of the previous slice, the current slice and the posterior slice (Supplementary Table 4) and the high recall mask and the high precision mask. The probability map for the view x-y, y-z and z-x are $\hat{p}_{xy}$, $\hat{p}_{yz}$ and $\hat{p}_{xz}$ respectively. And same with the combination function when segmenting the heart, the COVID-19 lesions and the lungs, we use $\hat{p}_{xy} + \hat{p}_{yz} + \hat{p}_{xz} > 2$ to cast the predictions into the binary segmentation mask.

1.4.4 The two-stage protocol dramatically improves the segmentation performance

In Algorithm 3 we defined the region discovery dice (RDD), and we found that the two-stage protocol dramatically improved the RDD of the blood vessel segmentations on the 148 CT scans with contrast agents. With the two-stage protocol, the scan-level RDD is 0.8518 ± 0.0268; while without the two-stage protocol, the scan-level RDD drops to 0.7667 ± 0.0293. Thus, the second-stage model significantly improves the RDD performance ($p < 0.0001$).

The improvements are mainly in tissues similar to blood vessels in the cross-section images. For example, pulmonary fissure and pulmonary nodules may be close to blood vessels while having similar CT signals with blood vessels. In these cases, only using the first-stage model cannot distinguish these difficult tissues, while the two-stage protocol consistently generates human comparable segmentations. For tiny regions, the two-stage protocol actually generates better segmentations. In Supplementary Figure 7 (d), there are at least 10 tiny regions

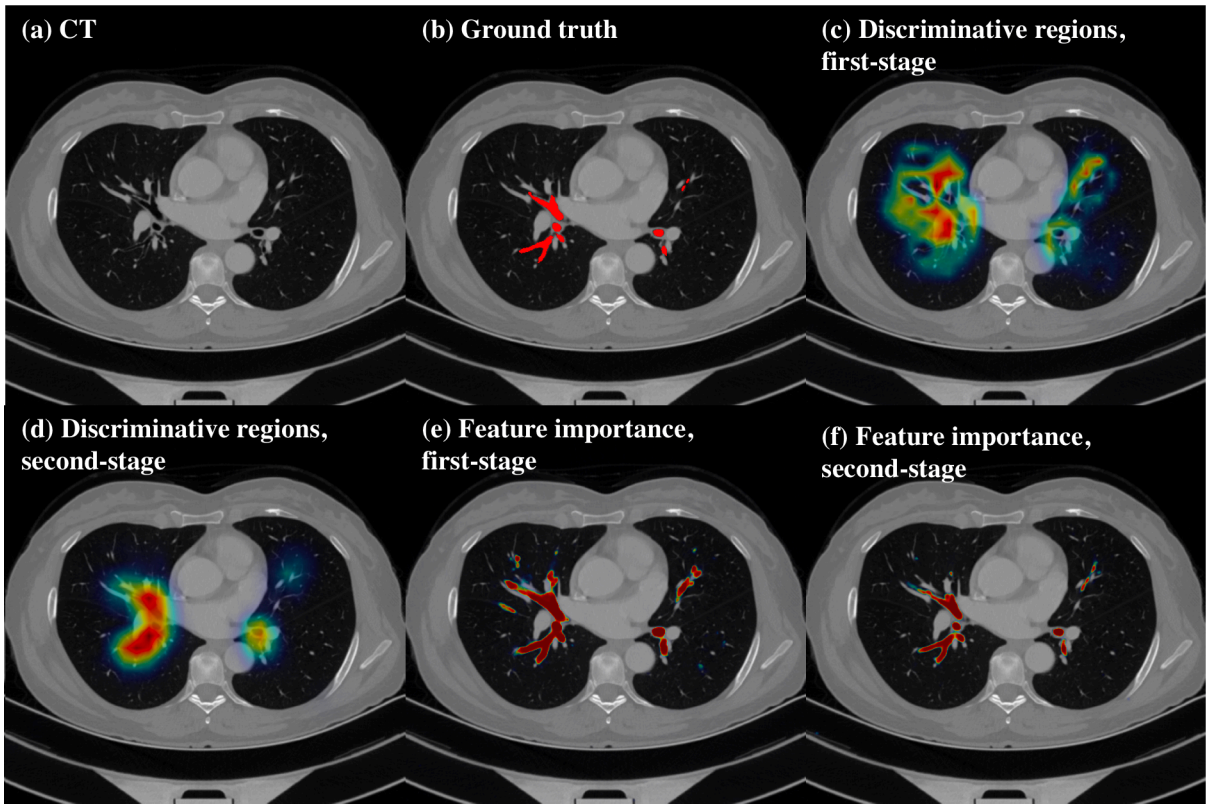
(green arrows, authenticated as true discoveries by radiologists) discovered by our methods but not shown in the ground truth, which is the main factor that our model cannot reach RDD close to 1 (in the slice shown in Supplementary Fig. 7, the RDD between the ground truth and the segmentation of two-stage segmentation is 0.7868 as our segmentation outperformed human annotations on the tiny blood vessels).



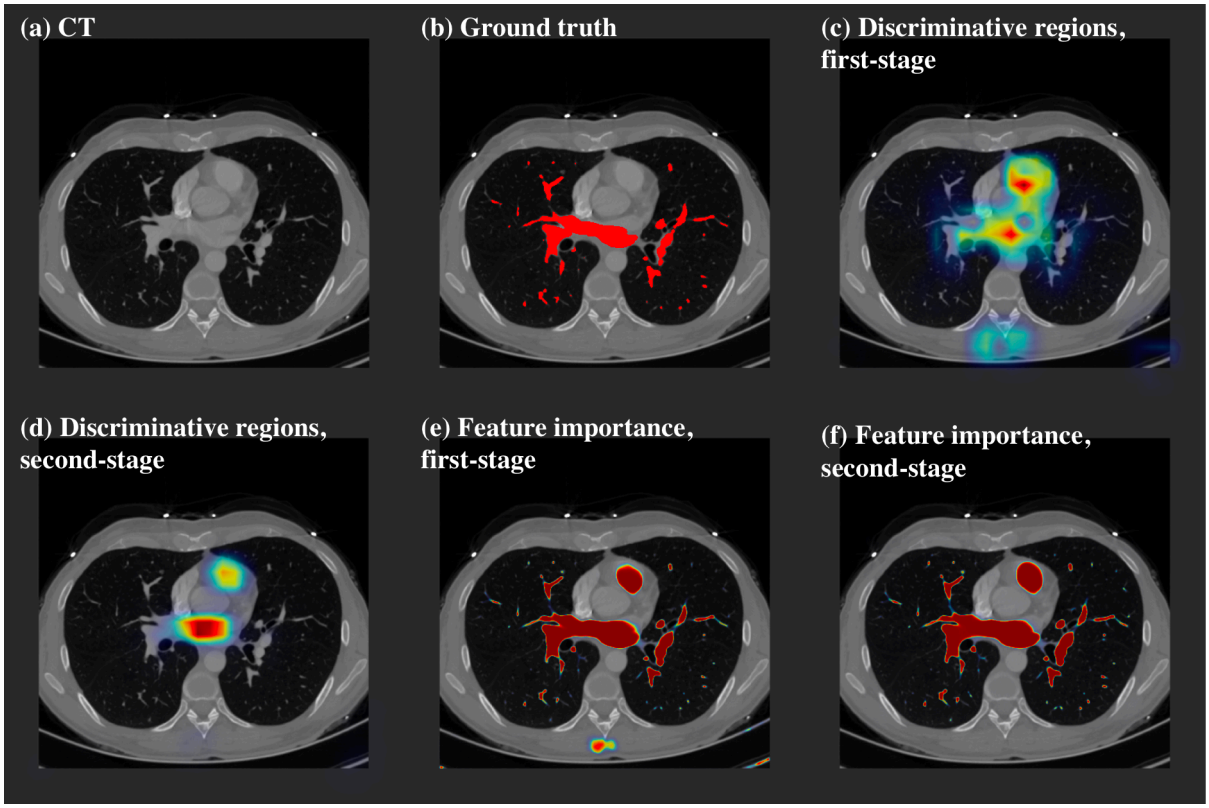
Supplementary Fig. 7 Ablation study of the two-stage protocol on the CT scans with contrast agents. The above figure shows an average case for the blood vessel segmentation. In this case, the region discovery dice (RDD) is 0.8580 while only use the first-stage model result in RDD of 0.7650. (a) Shows the spatial normalized CT slice from the x-y plane. (b) Red regions give the ground truth. (c) Red regions give the predictions of only using first-stage model. We can see that (the yellow arrow) the model miss-segmented the pulmonary fissure as blood vessels. (d) We can see with the two-stage protocol, the segmentation result is very close to the ground truth, while for tiny structures, the deep-learning model is actually better than human annotations (green arrows, authenticated as true discoveries by radiologists).

1.4.5 Visual interpretations

Here we give visual interpretations of how the two-stage protocol helps the model to focus on correct regions and thus improves the performances. Supplementary Figure 8 and Supplementary Figure 9 give the visual interpretations of the first-stage model and the second-stage model when segmenting airways and blood vessels. In these figures, the segmentations given by the two-stage protocol are human comparable. In general, the first-stage model looks for a wide range of regions including: the target semantic, backgrounds that provide important features, and structures that look like the target semantic; while the second-stage model focuses more on regions containing the target semantic, and the model will give no attention to structures that look like the target semantic. This explains why the two-stage protocol is good for the airway segmentation: in the large-scene-small-object problem, the first-stage model gives good guidance so that the second-stage model can focus on very small regions and generate human comparable segmentations.



Supplementary Fig. 8 Visual interpretation for the first-stage and the second-stage models when segmenting the airways. (a) The spatial rescaled CT from the x-y plane. (b) Red regions give the ground truth for the airways. (c) The discriminative regions for the first-stage model. We can see that when segmenting the airways, the model searches on a wide range of regions that may contain discriminative features. (d) The discriminative regions for the second-stage model. We can see that the second-stage model only searches on the regions that contain airways. (e) The feature importance map for the first-stage model. We can see that the model focused on the region broader than the airways, as the model needs to find out where the tracheal walls are. In addition, the first-stage model also focuses on some wrong regions. (f) The feature importance map for the second-stage model. We can see that the second-stage model only focused on the airway regions.

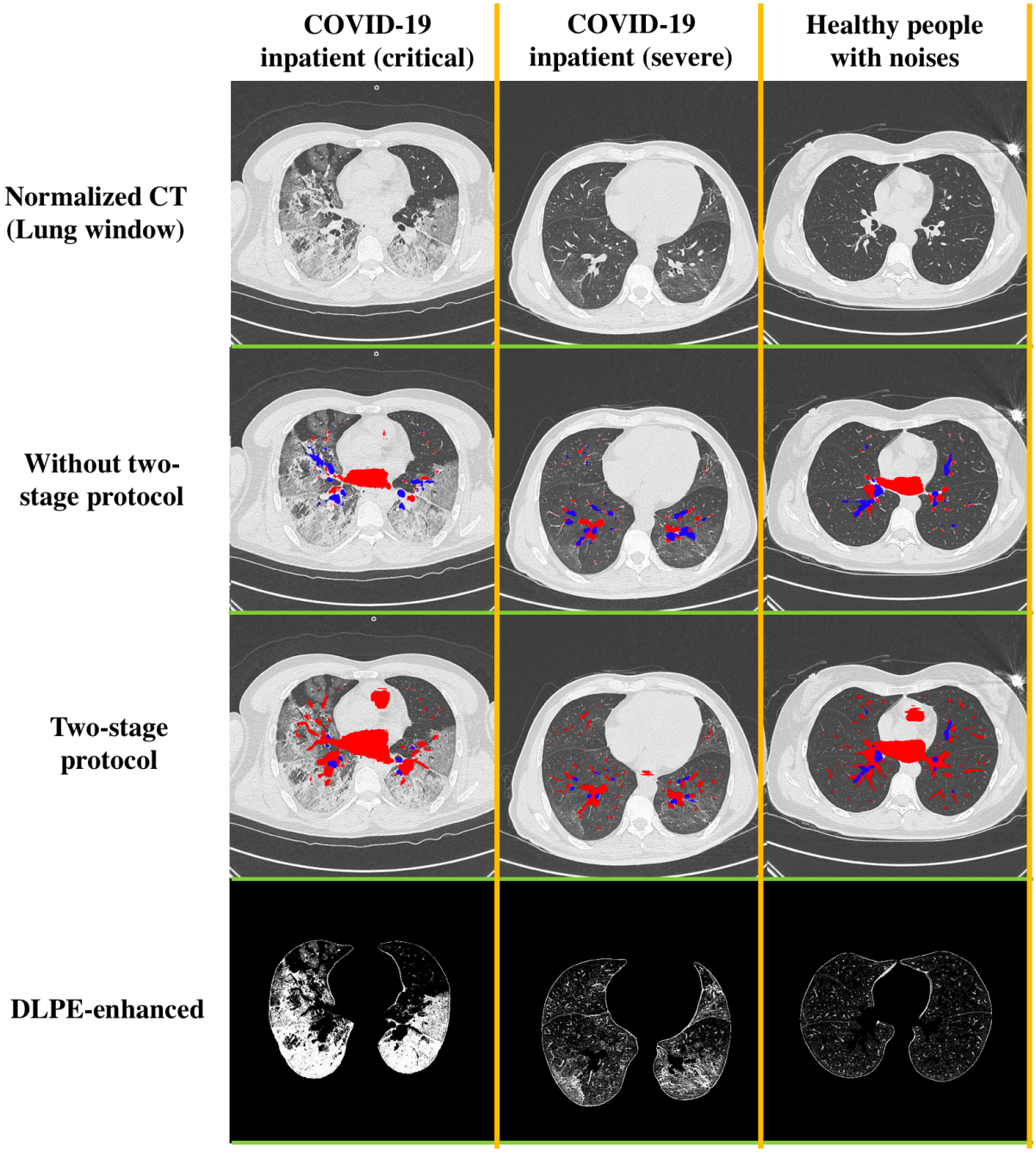


Supplementary Fig. 9 Visual interpretation for the first-stage and the second-stage models when segmenting the blood vessels. (a) The spatial rescaled CT from the x-y plane. (b) Red regions give the ground truth for the blood vessels. (c) The discriminative regions for the first-stage model. We can see the model looks on wide regions, e.g., the model searches on the bottom part where there are no blood vessels. (d) The discriminative regions for the second-stage model. We can see that the discriminative regions for the second-stage model are much more concentrated. (e) The feature importance map for the first-stage model. We can see that the model falsely focuses on the bottom regions. (f) The feature importance map for the second-stage model. We can see that the second-stage model only focused on the blood vessels, and put more focus on tiny vessels.

1.4.6 Robustness of the two-stage protocol and the DLPE methods

The DLPE method needs to be applied on chest CT scans from various patients, but the ground truth masks for airways and the blood vessels are from patients with no pneumonia. We find that the two-stage protocol dramatically improves the robustness of the airway and the blood vessel segmentations for the COVID-19 patients, especially for those patients with lots of lesions. Aided by the two-stage protocol, the model can segment the airways and blood vessels with high robustness, thus the DLPE method can generate stable and high quality parenchyma enhanced images.

In general, only using the first-stage model has difficulties when segmenting blood vessels and airways for CT scans with lesions and noise, while the two-stage protocol can consistently generate human comparable segmentations. For example, the chest CT of COVID-19 inpatients usually contain many lesions caused by inflammation, consolidations and fibrosis, and these lesions merge together with blood vessels and have very close CT signals with blood vessels or airways. Figure 10 gives three cases to give direct impressions about how the two-stage protocol improves the segmentation robustness. These cases do not have ground truth annotations, but radiologists highly agreed on our predictions and our predictions have natural and reasonable 3D morphologies. Based on the accurate segmentations, the DLPE enhanced images are clean and clear for these difficult cases.



Supplementary Fig. 10 The robustness of the two-stage protocol. The training set for airway and blood vessel segmentations do not contain CT scans with pneumonia or noise, and it is the two-stage protocol that enables stable and human comparable segmentations for these outliers. This means that the DLPE method has a high generalization power. Columns from left to right: COVID-19 critical inpatient; COVID-19 severe inpatient and healthy people with noises. Rows from top to bottom: spatially normalized CT from the lung window; the airway segmentation (blue regions) and the blood vessel segmentation (red regions) without the two-stage protocol; the segmentations (blue for airways and red for blood vessels) with the two-stage protocol; and the effects after the DLPE enhancement.

2 Applying DLPE on COVID-19 Inpatients and Survivors

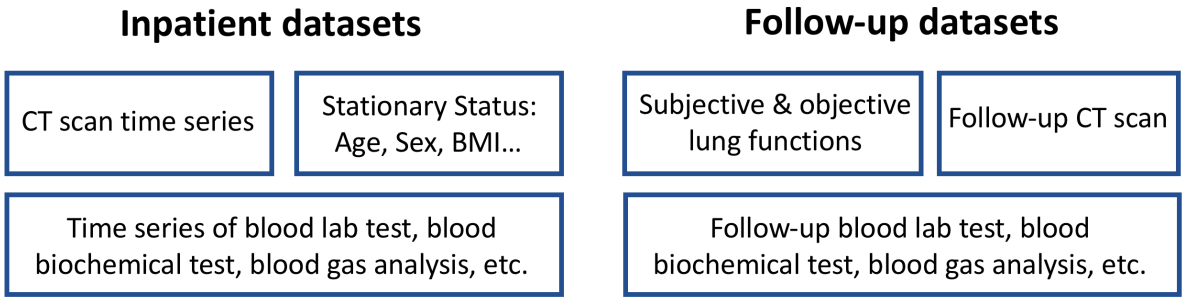
DLPE aided us to find novel lesions, and this part describes the methods in order to analyze the origins and the properties of these novel lesions.

2.1 Dataset preparation

2.1.1 Data collection

The inpatient and the follow-up datasets are from the same group of people (46 people) who were severe or critical COVID-19 patients. Supplementary Figure 11 gives the overview of our datasets. Each patient in the inpatient datasets has collected a time serie of CT scans, time series of clinical metrics and been recorded stationary status like age, sex, etc. A tremendous number of clinical metrics were collected if the patient was in the intensive care unit (ICU). For example, every day, the blood tests collected more than 400 clinical metrics for ICU patients.

The patients in the inpatient datasets came back months after discharge, and formed the follow-up datasets. Each survivor in the follow-up datasets has collected the CT scan, subjective lung functions, objective lung functions and clinical metrics including blood tests, blood gas analysis, etc.



Supplementary Fig. 11 Overview of the inpatient datasets and the follow-up datasets. These datasets are from the same group of people (46 people) who were severe or critical patients. During the hospitalization, we collected their CT scan and clinical metrics every day or every few days, and recorded the stationary status like age, sex, medical history for the patients. These patients came back and we recorded their lung functions, CT scans and a variety of clinical metrics.

2.1.2 Extract features from time series in the inpatient datasets

We need to extract temporal features from the time series of clinical metrics. Most time series contain 2 to 5 timepoints, so we decide to use two features to depict the time series: the average value and the extreme difference. The extreme difference (the maximum subtracts the minimum, and takes the negation if the maximum occurs earlier than the minimum).

Thus, each time serie will be cast into two temporal features: “average” and “extreme”. If the number of timepoints for a time serie is zero (patient did not collect this metric), the “average” and “extreme” will be marked as “None”. If the number of timepoints for a time serie is one, the “extreme” will be marked as “None” (see Algorithm 5 for details).

Radiomics are calculated by lesions, and we know that more lesions implies worse conditions. Thus, we only record the maximum of the inpatient radiomics, which depict the most difficult period of a COVID-19 patient.

Algorithm 5: Calculate temporal features

Inputs: time serie
 # The time serie of a metric. In our datasets, most time series with the length of 3 to 5.
Output: temporal features
 # The outputs are floats or None

```
1. if len(time serie) == 0:
2.         return None, None
3. average = mean(time serie)
```

 # calculate the average value

4. **if** len(time serie) == 1:
5. **return** average, **None**
6. extreme = max(time serie) − min(time serie)
7. **if** index of max(time serie) > index of min(time serie):
8. extreme = − extreme
9. **return** average, extreme

2.1.3 Feature reduction

Each patient in the datasets has 659 features, including inpatient temporal features, stationary status, follow-up lung functions, follow-up CT, follow-up blood tests.

Many of the features are redundant and can be neglected: many of the features (e.g., antibody for herpes) are not likely to have any correlation with COVID-19 symptoms and sequelae; many of them (e.g., different measurements for the same metric) have very strong correlations; and many of them are not key metrics and were only collected by very few people.

We discard a feature if more than 40% of the people in the datasets do not have this feature, because important metrics should be collected by most of the people. We discard a feature if the feature values for all people are always in the normal range. These two steps reduce the number of features from 659 to 228.

We discard a feature if it is unlikely to have any correlation with COVID-19 symptoms and sequelae. These features are usually antibodies and traits for other diseases. This step reduces the number of features from 228 to 209.

Medical experts discard features that are of the same metric but collected by different measurements. For example, the PaO2/FiO2 ratio is approximated non-invasively many times a day, but to test the ground truth PaO2/FiO2 ratio needs the artery blood and is tested every day or every two days. Thus, the temporal features of the approximated PaO2/FiO2 ratio are discarded. This step reduces the features from 209 to 189.

Finally, each patient in the datasets has 189 features, including inpatient temporal features, stationary status, follow-up lung functions, follow-up CT and follow-up blood tests.

2.1.4 Data normalization

We use min-max normalization for all features. The feature values are normalized by:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{27}$$

Here x is the original feature value, x_{min} (x_{max}) is the minimum (maximum) value of this feature among all people, and x' is the feature value after min-max normalization.

2.1.5 Target features

Some features are important for COVID-19 patients, thus, we explore how other factors can predict these important features. In the inpatient dataset, the PaO2/FiO2 ratio (PFR) is the most important metric for the respiratory functions of COVID-19 inpatients, and we use PFR to classify the mild, severe and critical patients. Thus, we want to explore the relationships between PFR and other inpatient features.

In the follow-up dataset, nearly all survivors reported significant respiratory sequelae. Thus, we analyze factors that cause these sequelae. We have 16 respiratory sequelae, which are listed in Supplementary Table 7.

In all, the target features are PFR and the 16 respiratory sequelae. We will call them “targets”.

Supplementary Table 7 Respiratory sequelae for COVID-19 survivors. First column, the abbreviation of the sequelae. Second column, the full name of the sequelae. Third column, brief descriptions of the sequelae.

Abbreviation	Full name	Description
mMRC	Modified medical research council	A quick self-rating scale to measure breathlessness
SGRQ	St. George's respiratory questionnaire	A comprehensive self-rating scale for the life quality of patients with chronic obstructive pulmonary diseases

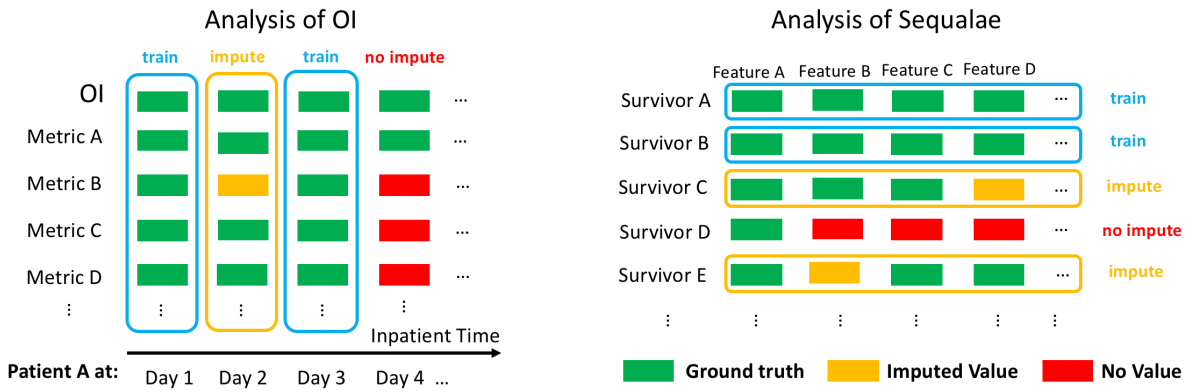
DLCO/VA	Diffusing capacity for carbon monoxide to alveolar volume ratio	Measures the capability of the lungs to transfer gas into capillaries
pO2(A-a)	Alveolar-arterial oxygen tension difference	Measures the efficiency of oxygen exchange in the lungs
FEV1	Forced expiratory volume in one second	The maximum air volume that can be exhaled in one second
FVC	Forced vital capacity	The maximum air volume that is exhaled during fast exhaling
FEV1/FVC	FEV1 to FVC ratio	Less than 0.7 indicates airflow limitations
MEF50	The maximal expiratory flow at 50 % of the forced vital capacity	Reduced MEF50 indicates small airway disease
TLC	Total lung capacity	The maximum gas volume inside the lungs
FRC	Functional residual capacity	Volume of gas in the lungs after calm exhalation
RV	Residual volume	Volume of gas in the lungs after maximum exhalation.
ERV	Expiratory reserve volume	The maximum volume that can be exhaled after calm exhalation
IC	Inspiratory capacity	The maximum gas volume that can be inhaled after calm exhalation
PEF	Peak expiratory flow	Maximum flow when exhaling forcefully
VT	Tidal volume	The amount of air inhaled or exhaled during calm breathing
MV	Minute ventilation	The volume of gas inhaled by the lungs per minute.

2.1.6 Data imputation

We use the KNN imputer to impute the missing features. The KNN imputer trains a mapping from nearby features to the target feature that contains missing values. Thus, before data imputation, we place features and metrics of the same aspect into adjacent columns or rows as they usually have high correlations. For example, before imputing data for sequelae prediction, we classify the features: column 1-2 are subjective lung functions; column 3-16 are objective lung functions; column 16-28 are radiomics; column 29-34 are stationary status; column 35-60 are follow-up blood lab tests, etc.

Note that the data imputation requires that the adjacent features or metrics can predict the missing value, thus, if a patient misses all values of an aspect, e.g., did not complete the blood-gas analysis, we will not impute values for the aspect.

Supplementary Figure 12 illustrates the data imputation process. When predicting PFR, we use the metrics collected on the same day to predict the PFR on that day. As shown in the left figure, if there exist missing values, e.g., metric B on the day 2, the imputer uses other columns to train the mapping to impute the missing value. However, if it misses all values of an aspect, like on day 4, we will not impute it and day 4 will not become a sample in further analysis.



Supplementary Fig. 12 The data imputation process for the analysis of PFR and respiratory sequelae. Green block means that the metric or feature is collected. Yellow block means the imputer impute the missing value. Red block means that the imputer cannot impute the missing value. Left panel: imputation for the prediction of PFR. We use the metrics collected on the same day to predict the PFR. If there exists missing values, e.g., metric B on the day 2, the imputer uses other columns to train the mapping to impute the missing value. However, if it misses all values of an aspect, like on day 4, we will not impute it and

day 4 will not become a sample in further analysis. Right panel: imputation for the prediction of respiratory sequelae. There are 7 patients who did not impute, because they did not have metrics of the blood-gas analysis.

2.2 The sample inclusion and exclusion criteria

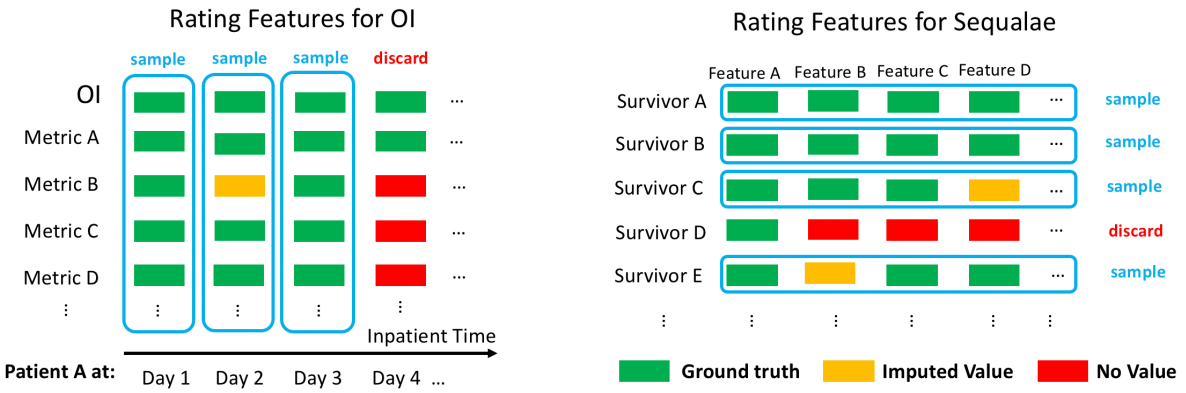
This section describes the inclusion and exclusion criteria for Supplementary Section 2.3 and 2.4.

The data analysis has two stages. The first stage is to select informative features (Supplementary Section 2.3) from hundreds of candidate features, and the second stage is to analyze the relationships between informative features and the target features (Supplementary Section 2.4).

2.2.1 During feature rating

There are hundreds of features, but many of them do not contain much information. The feature rating algorithms analyze how these hundreds of features relate to the targets, and pick out features containing information (see Supplementary Section 2.3 for feature rating algorithms).

We discard samples containing missing values that cannot be imputed, and remain all other samples.



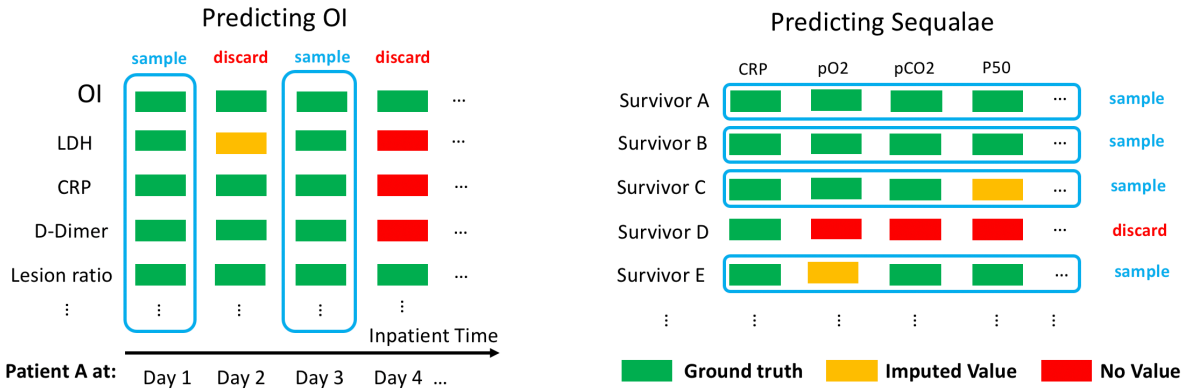
Supplementary Fig. 13 The inclusion and exclusion criteria for samples during the feature rating. We discard samples containing missing values that cannot be imputed, and remain all other samples.

2.2.2 During regression

The feature rating algorithms rated out informative features with the best discriminative powers, and we then use these features to predict PFR and respiratory sequelae.

For the prediction of PFR, each patient provides several samples as the inpatient time can be as long as one month. Thus, we only include samples that all values are ground truth, and discard samples with imputed values or missing values.

A large number of features contain information for the predictions of respiratory sequelae, thus, we keep samples with imputed value and only discard samples containing missing values that cannot be imputed.



Supplementary Fig. 14 The inclusion and exclusion criteria for samples during regression. We have lots of inpatient samples for predicting PFR, thus, we only include samples that all values are ground truth. On the other hand, there are many features containing information for the sequelae prediction, thus, we include samples with imputed value.

2.3 Feature rating and selection

2.3.1 Aim and idea of the feature selection

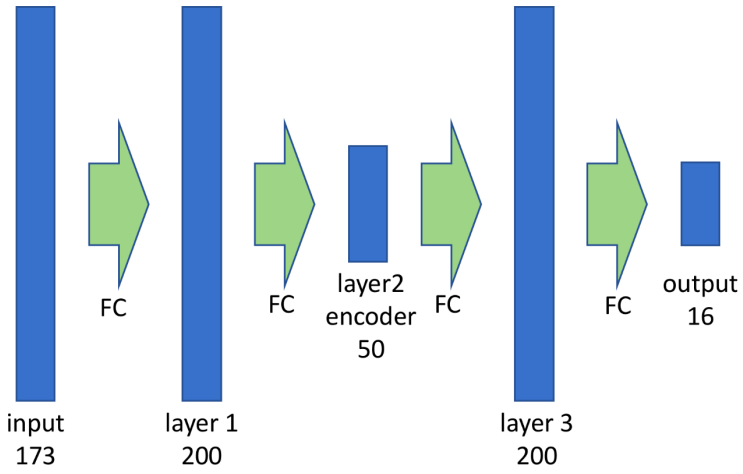
The features selected in this section will be the input features in Supplementary Section 2.4. There are hundreds of features that may have predictive power to the target features. This section aims to pick out features that contain information, which reduce the features from hundreds to dozens.

We compare the discriminative powers of these features with random noise, and only features containing more information than random noise with $p < 0.001$ will be kept. Among the remaining features we select out the most informative ones, and the features selected out constitute 95% of the total discriminative power.

Methods like multi-variable analysis and XGBoost can rate the input features based on their discriminative powers. Multi-variable analysis should be better than XGBoost during the variable selection due to two reasons. First, XGBoost tends to give higher ranks to features that are fed into the model earlier: XGBoost builds the decision tree based on the features, but the final decision tree and feature importance ranking are influenced by the order of features fed into XGBoost. By comparison, the multi-variable analysis does not have this problem. Second, XGBoost is only suitable for one target feature, but we have 16 respiratory sequelae. Thus, XGBoost cannot select features for the prediction of respiratory sequelae.

2.3.2 The multi-variable analysis model

We use a fully connected neural network to establish the mapping between features and targets. During training, the network will put more focus on targets that can be predicted, and give more weights to the features containing more information. Finally, we can get the feature ranking that indicates the amount of information of these features, and the target ranking indicates the predictability of these targets.



Supplementary Fig. 15 The architecture of the fully connected network when predicting respiratory sequelae. There are 173 inpatient and follow-up features and 16 targets (respiratory sequelae). "FC" means fully-connected.

Take the prediction of respiratory sequelae as an example. Denote $\boldsymbol{x}$ as the input features, if the batch size is one, then $\boldsymbol{x}$ is shaped 1×173. Denote $\boldsymbol{y}$ as the ground truth target values, which is shaped 1×16. Denote $\boldsymbol{y}'$ as the predicted target values, which is shaped 1×16. Denote the feature rating as $\boldsymbol{r}_{in}$, which is a vector shaped 1×173, indicating the amount of information of each input feature. Denote the target rating as $\boldsymbol{r}_{out}$, which is a vector shaped 1×16, indicating the predictability of each output target. We require that $\mathrm{sum}(\boldsymbol{r}_{out})$ and $\mathrm{sum}(\boldsymbol{r}_{in})$ are always one. Initially, values in $\boldsymbol{r}_{in}$ are all $\frac{1}{173}$ while values in $\boldsymbol{r}_{out}$ are all $\frac{1}{16}$.

The input of the model is the element-wise multiplication of $\boldsymbol{r}_{in}$ and $\boldsymbol{x}$, or $\boldsymbol{x} \circ \boldsymbol{r}_{in}$.

The loss function L has two parts: the regression loss and regularization loss.

$$L = \mathrm{sum}((\boldsymbol{y} - \boldsymbol{y}')^2 \boldsymbol{r}_{out}^T) + \beta L_{reg} \tag{28}$$

We set $\beta = 0.01$ in our study. L_{reg} is the L2 regularization loss, which calculates the average value for the square of the learnable weights.

$$L_{reg} = \mathrm{mean}(weight^2) \tag{29}$$

The error vector is defined as:

$$\boldsymbol{D} = \sum_i (\boldsymbol{y}_i - \boldsymbol{y}'_i)^2 \circ \boldsymbol{r}_{out} \tag{30}$$

Here i means the i-th patient. $\boldsymbol{D}$ is a vector shaped 1×16. We use the five-fold CV.

We need to train the model to converge with given r_{in} and $\boldsymbol{r}_{out}$ (need 200-300 epochs with learning rate 0.01 and Adam Optimizer and), and then update $\boldsymbol{r}_{in}$ and $\boldsymbol{r}_{out}$.

2.3.3 Update feature rating

r_{in} indicates the relative information amount for input features, and we give higher weights for features containing more information. Denote σ as the standard deviation of the k-th input feature. We replace the feature values with Gaussian noise of $\mathrm{Normal}(0, \sigma)$, which removes the information from the k-th feature. Thus, the error $\mathrm{sum}(\boldsymbol{D})$ is likely to increase, and the increase measures the relative information of the k-th feature. This process will be rerun for 10,000 times for better evaluation, see Algorithm 6. As shown in the algorithm, we require the feature to outperform the random noise with probability higher than $p > 0.999$, otherwise the feature will be discarded.

Algorithm 6: Calculate the relative information

Inputs: dataset, k, model

 # Here the "k" means we are calculating the k-th feature.

 # Here the model is trained to converged with given $\boldsymbol{r}_{in}$ *and* $\boldsymbol{r}_{out}$.

Output: relative information for the k-th feature

 # The outputs is a float

1. D = model(dataset):

 # Here D is defined in equation (30)

2. increase = 0
3. count worse than random = 0

 # Record the number of times the random noise outperforms the feature

4. **for** i **in** range(10000)**:**
5. replace the *k*-th feature with noise
6. error increase = sum(model(dataset with noise)) − sum(D)
7. **if** error increase < 0**:**

this means the noise seems contains more information than the feature

8. count worse than random += **1**
9. **if** count worse than random > 10**:**
10. **return** 0
11. increase = increase + error increase
12. **return** max(increase, 0)

Thus, we can calculate the relative information for each feature. Denote the relative information for the k-th feature as I_k, then we update the feature rating $\boldsymbol{r}_{in}$ according to I_k:

$$\boldsymbol{r}_{in} = \frac{(I_1, I_2, I_3, \dots, I_{173})}{\sum_k I_k} \tag{31}$$

Once $I_k = 0$ during any updating, I_k will be fixed to zero forever. Thus, only features that outperform random noise ($p < 0.001$) will be considered as informative.

2.3.4 Update target rating

$\boldsymbol{r}_{out}$ indicates the relative predictability for the targets, and we want the model to put more focus on targets that are predictable. Initially, each input $\boldsymbol{x}_i$ corresponds to a ground truth $\boldsymbol{y}_i$, and we have the error vector $\boldsymbol{D}$. During the target rating, we shuffle the correspondence between $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ then retrain the model, which erases the predictability of all targets. Thus, the error is likely to increase for each target, and the increase measures the predictability of the target. We redo this process for ten times and get the error vectors for the shuffled models $\boldsymbol{D}_1, \boldsymbol{D}_2, \dots, \boldsymbol{D}_{10}$.

The original error vector is $\boldsymbol{D}$ shaped 1×16, and $\boldsymbol{D}[k]$ is the error for the k-th target. The relative predictability for the k-th target is defined as:

$$Z_k = \text{Relu}\left(\frac{\sum_{j=1}^{10}\left(\boldsymbol{D}_j[k] - \boldsymbol{D}[k]\right)}{\sum_{j=1}^{10} \boldsymbol{D}_j[k]}\right) \tag{32}$$

Here $\sum_{j=1}^{10} \boldsymbol{D}_j[k]$ is the total error for the shuffled models, which quantifies the error if the k-th target is totally unpredictable. $\sum_{j=1}^{10}\left(\boldsymbol{D}_j[k] - \boldsymbol{D}[k]\right)$ quantifies how much error is explained. Thus, we update the target rating to:

$$\boldsymbol{r}_{in} = \frac{(Z_1, Z_2, \dots, Z_{16})}{\sum_k Z_k} \tag{33}$$

2.3.5 The rating workflow

Start with initial $\boldsymbol{r}_{in}$ and $\boldsymbol{r}_{out}$, we train the model, then update $\boldsymbol{r}_{in}$ and $\boldsymbol{r}_{out}$ iteratively. $\boldsymbol{r}_{in}$ and $\boldsymbol{r}_{out}$ converge after 4-7 iterations. For simplicity, we stop updating when $\max(\boldsymbol{r}_{out})$ is converged. Algorithm 7 gives the workflow for rating the features and targets during the prediction of the respiratory sequelae.

Algorithm 7: Rating workflow for respiratory sequelae prediction

Inputs: dataset

Output: $\boldsymbol{r_{in}}, \boldsymbol{r_{out}}$, model

 # The ratings and the final model

1. $\boldsymbol{r_{in}}$ = (1/173, 1/173, …)
2. $\boldsymbol{r_{out}}$ = (1/16, 1/16, …)

 # Initially, all weights are the same

3. **while** max($\boldsymbol{r_{out}}$) is not converge**:**
4. model = training($\boldsymbol{r_{in}}$, $\boldsymbol{r_{out}}$, dataset)
5. $\boldsymbol{r_{in}}$ = update_input_rating(model, $\boldsymbol{r_{out}}$, dataset)
6. $\boldsymbol{r_{out}}$ = update_output_rating(model, $\boldsymbol{r_{in}}$, dataset)
7. **return** $\boldsymbol{r_{in}}$, $\boldsymbol{r_{out}}$, model

Based on the workflow, we get the rating for the discriminative powers of the features, the rating for the predictability of the targets, and the model that builds the mapping between input features and the targets.

2.3.6 Comparisons between XGBoost

We also apply Algorithm 7 to rate inpatient features that predict the PaO2/FiO2 ratio, and we skip the line 6 of Algorithm 7 as there is only one target.

As there is only one target, XGBoost can also rate the importance of the input features. We compare the feature ranking list given by XGBoost with $\boldsymbol{r}_{in}$ given by Algorithm 7. We find that they conform with each other. 1) The three most important features are the same, which are the lesion ratio of all parenchyma, lactate dehydrogenase (LDH), and C-reactive protein (CRP). 2) The 10 most important features have 6-8 overlaps (the ratings of the XGBoost are slightly influenced by the order of features that is fed into XGBoost).

Thus, feature rating by the multi-variable analysis and XGBoost have high conformity, which means that $\boldsymbol{r}_{in}$ should give satisfactory ratings for the features.

2.3.7 Criterion for the feature selection

We select features with the most information according to $\boldsymbol{r}_{in}$, and the selected features contain 95% of the total information. See Algorithm 8.

Algorithm 8: Feature selection

Inputs: $\boldsymbol{r_{in}}$
Output: list of feature names for the selected features

1. total information = 0
2. list_feature_selected = []
3. **while** total information < 0.95 **:**
4. name, value = $\boldsymbol{r_{in}}$.pop_max()

 # pop the feature name with the most information.

5. list_feature_selected.append(name)
6. total information += value
7. **return** list_feature_selected

2.3.8 Features selected for predicting PFR

For the prediction of PFR, the number of original inpatient features is 147, and we select 3 radiomics, 6 inpatient clinical metrics and 3 basic information features, which are listed in Supplementary Table 8.

The 6 inpatient clinical metrics are: lactate dehydrogenase (LDH), C-reactive protein (CRP), lymphocyte absolute count (lym_abs), neutrophils absolute count (neu_abs), neutrophils to lymphocyte ratio (neu_to_lym), and D-Dimer.

The 3 basic information features are: sex, age, and body mass index (BMI).

The radiomics are: R1, R2 and the total signal difference between lesions and baseline (R3). In order to compare the differences with and without DLPE, the lesions will be calculated by state-of-the-art COVID-19 lesion quantification methods as well as the DLPE method.

Supplementary Table 8 The 12 selected inpatient metrics for predicting PFR.

Abbreviation	Full name	Abbreviation	Full name
LDH	Lactate dehydrogenase	Sex	Sex
CRP	C-reactive protein	Age	Age at hospitalization
Lym_abs	Lymphocyte absolute count	BMI	Body mass index at hospitalization
Neu_abs	Neutrophils absolute count	R1	The median signal difference between lesions and baseline
Neu_to_lym	neutrophils to lymphocyte ratio	R2	The ratio between the lesion volume and the lung volume
D-Dimer	Blood D-Dimer concentration	R3	The total signal difference between lesions and baseline

2.3.9 Features selected for predicting respiratory sequelae

For the prediction of the 16 respiratory sequelae, the number of original features is 173, and we select out 53 features: the 21 inpatient features for clinical metrics and 3 radiomics during hospitalization, 5 basic information features, 6 follow-up CT radiomics and 18 follow-up lab tests. These features are listed in the Supplementary Table 9.

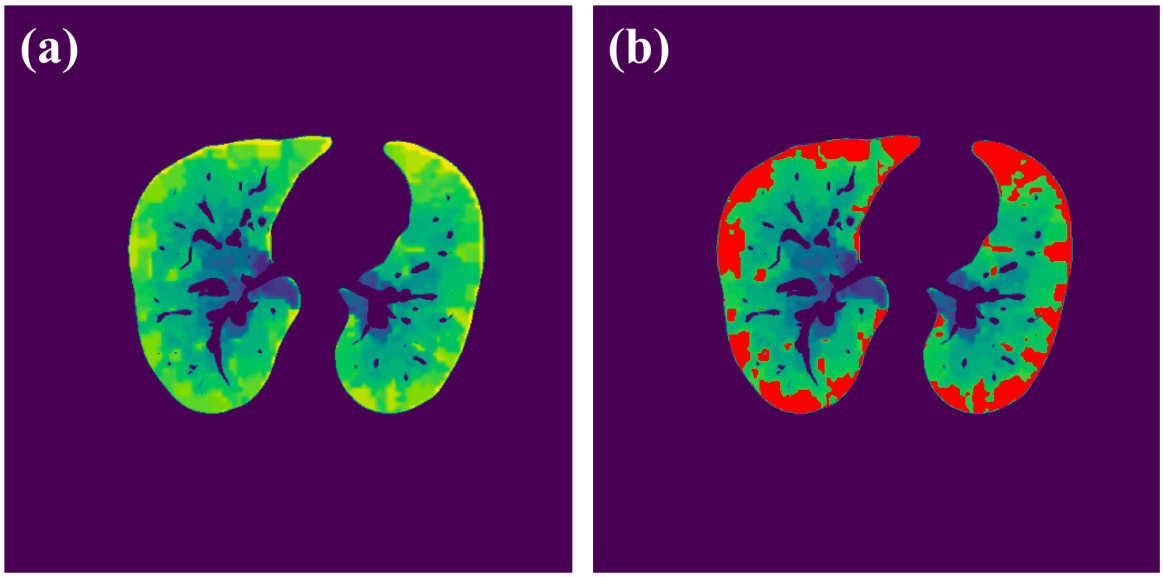
Note that for inpatient time series, we extract the “average” and the “extreme” features, and the 21 inpatient features for clinical metrics are: interleukin-2 (IL-2) average, CRP average, interleukin-4 (IL-4) average, tumor necrosis factor (TNF) average, activated partial thromboplastin time (APTT) average, erythrocyte sedimentation rate (ESR) average, mean corpuscular volume (MCV) average, IL-2 extreme, interferon gamma (IFG) average, the average of aspartate transaminase (AST) to alanine aminotransferase (ALT) ratio, D-Dimer average, thrombin time (TT) average, gamma-glutamyl transferase (GGT) average, eosinophil count (EC) average, hemoglobin (Hb) average, eosinophil ratio (ER) average, thrombocytes count (TC) average, sodium average, LDH average, albumin average, and creatine kinase (CK) average.

The average of inpatient radiomics are all rated out by the algorithm, they are the average for: the median lesion severity (R1 average), the lesion volume ratio (R2 average), and the total lesion severity (R3 average). Here the lesion is quantified by the DLPE method, as we find that DLPE quantifies better radiomics.

Five basic features include: sex, age, BMI, height, and weight.

All six follow-up radiomics are rated out, they are: the median lesion severity (R1), the lesion volume ratio for total parenchyma (R2), the total lesion severity (R3) for total parenchyma, and the R1, R2, R3 for the lower respiratory regions. We will use “R1 total” and “R1 lower” to distinguish the total parenchyma and the lower respiratory regions. The lower respiratory regions are illustrated by Supplementary Figure 16. As most follow-up lesions are invisible under the lung window, the follow-up lesions are quantified by the DLPE method.

The 18 follow-up blood tests rated out are: LDH, blood oxygen partial pressure (pO2), D-Dimer, blood lactate concentration (cLac), oxygen partial pressure at 50% blood oxygen saturation (P50), change in total bilirubin (ctBil), CRP, blood carbon dioxide partial pressure (pCO2), CK, GGT, prothrombin time (PT), hemoglobin (Hb), leucocytes count (Leu), fractional saturation (FO2Hb), ALT, plasma creatinine (Cr), blood urea nitrogen (BUN), and blood pH value.



Supplementary Fig. 16 Illustration of the lower respiratory regions. We use the branching level of the nearest blood vessels to approximate the lower respiratory regions. **(a)** Branching level of the nearest blood vessels. Brighter means higher branching level, and the brightest is of branching level around 12. **(b)** Red regions give the estimated lower respiratory regions. If the nearest blood vessel is of branching level > 7 for a parenchyma voxel, we will classify the voxel as a lower respiratory voxel.

Supplementary Table 9 The 53 selected features for the predictions of respiratory sequelae. If the abbreviation with "average" or "extreme", this means that the feature is extracted from inpatient time series, and we will neglect the "average" or "extreme" in the column of full name. For the follow-up radiomics, we use "total" and "lower" to distinguish whether the lesions are calculated from the total parenchyma or the lower respiratory regions. If the full name for a follow-up metric coincides with the inpatient feature, we will add "follow-up".

Abbreviation	Full name without "average" or "extreme"	Abbreviation	Full name
IL-2 average	Interleukin-2	Height	Height at follow-up
CRP average	C-reactive protein	Weight	Weight at follow-up
IL-4 average	Interleukin-4	R1 total	Median follow-up lesion severity for all parenchyma
TNF average	Tumor necrosis factor	R2 total	Follow-up lesion ratio for all parenchyma
APTT average	Activated partial thromboplastin time	R3 total	Follow-up total lesion severity for all parenchyma
ESR average	Erythrocyte sedimentation rate	R1 lower	Median follow-up lesion severity for all parenchyma
MCV average	Mean corpuscular volume	R2 lower	Follow-up lesion ratio for all parenchyma
IL-2 extreme	Interleukin-2	R3 lower	Follow-up total lesion severity for all parenchyma
IFG average	Interferon gamma	LDH	Lactate dehydrogenase
AST/ALT average	Aspartate transaminase to alanine aminotransferase ratio	pO2	Blood carbon dioxide partial pressure
D-Dimer average	D-Dimer	D-Dimer	D-Dimer follow-up
TT average	Thrombin time	cLac	Blood lactate concentration
GGT average	Gamma-glutamyl transferase	P50	Oxygen partial pressure at 50% blood oxygen saturation
EC average	Eosinophil Count	ctBil	Change in total bilirubin
Hb average	Hemoglobin	CRP	C-reactive protein follow-up
ER average	Eosinophil ratio	pCO2	Blood carbon dioxide partial pressure

TC average	Thrombocytes count	CK	Creatine kinase follow-up
Sodium average	Blood sodium concentration	GGT	Gamma-glutamyl transferase follow-up
LDH average	Lactate dehydrogenase	PT	Prothrombin time
Albumin average	Albumin	Hb	Hemoglobin follow-up
CK average	Creatine kinase	Leu	Leucocytes count
R1 average	Median lesion severity for all parenchyma	FO2Hb	Fractional saturation
R2 average	Lesion ratio for all parenchyma	ALT	Alanine aminotransferase
R3 average	Total lesion severity for all parenchyma	Cr	Plasma creatinine
Sex	Sex	BUN	Blood urea nitrogen
Age	Age at hospitalization	pH	Blood pH value
BMI	Body mass index at follow-up		

2.4 Detailed results for predicting respiratory sequelae

2.4.1 The prediction method

The multi-variable analysis (or neural network) model, Lasso and XGBoost are commonly used regression methods. We find that these methods conform to each other to a great extent, and XGBoost outperforms others by a small extent in most cases. Thus, we use the XGBoost to do the prediction for all targets (PFR, mMRC, SGRQ, etc.).

We use the leave-one-out cross-validation to evaluate how the input can predict the target feature. When predicting PFR, the inputs are the 12 selected features (among them there are 3 radiomics), and the output is the predicted value for PFR. When predicting the respiratory sequelae, the inputs are the 53 selected out features (among them there are 6 radiomics), and the output is the predicted value for a sequela (16 sequelae means we need to train 16 XGBoost models).

The XGBoost can rank the input features according to the discriminative power. Despite the ranking being slightly influenced by the order of features that are fed into the XGBoost model, we find that the top three most informative features are always the same.

2.4.2 The prediction, feature ranking and ablations with XGBoost

Supplementary Table 10 gives the prediction, the feature ranking and the ablation studies for each target. In general, the DLPE method plays a crucial role for extracting features to predict the gas transferring capabilities (DLCO/VA, pO2(A-a)), the lung capacities (FVC, TLC), and the comprehensive life quality scale SGRQ. In these cases, the PCC will drop significantly without information provided by DLPE. In addition, the PCC without DLPE decreases in most cases, which means the radiomics quantified by DLPE should provide information for many scenarios.

Supplementary Table 10 Using inpatient and follow-up clinical metrics and radiomics to predict the COVID-19 long-term respiratory sequelae. See Supplementary Table 7 for the descriptions of these sequelae (the first column). See Supplementary Table 9 for the description of these targets (the third column). The prediction model is XGBoost. First column, the 16 sequelae. Second column, the PCC between the predicted value and the ground truth value. Third column, the top three most informative features ranked by the XGBoost. Fourth column, replace the radiomics with the lesion quantification of the state-of-the-art methods for COVID-19 lesions, but without using the DLPE scheme, and re-train the model. Fifth column, radiomics that without DLPE to remove bias will significantly decrease the PCC: * means $\boldsymbol{p < 0.01}$, ** means $\boldsymbol{p < 0.001}$ and *** means $\boldsymbol{p < 0.0001}$. We compare the second column and the fourth column, and the best performer is in bold.

Target	PCC with DLPE	Top three informative	PCC without DLPE	DLPE Sensitive Radiomics
SGRQ	**0.732**	R2 total, R1 total, R1 lower	0.243	R2 total***, R1 total**
DLCO/VA	**0.550**	ctBil, R2 Total, pH	0.327	R2 total*
pO2(A-a)	**0.718**	Leu, ctBil, R2 total	0.532	R2 total*
FEV1	**0.623**	ctBil, FO2Hb, R1 lower	0.471	R1 lower*
FVC	**0.605**	R2 total, ctBil, cLac	0.325	R2 total**, R2 average*

FEV1/FVC	**0.649**	R1 lower, FO2Hb, CK	0.387	R1 lower**
MEF50	**0.462**	CRP, pO2, PT	0.411	None
TLC	**0.705**	R1 total, pO2, R1 lower	0.391	R1 total**, R1 lower*, R2 lower*
FRC	**0.434**	R2 total, cLac, CK	0.395	None
RV	0.413	P50, ALT, Age	**0.420**	None
ERV	**0.729**	Height, R2 total, Cr	0.546	R2 total*
IC	0.444	CRP, ctBil, Leu	**0.449**	None
PEF	**0.450**	pO2, ALT, R1 lower	0.375	None
VT	**0.386**	BMI, Cr, R1 lower	0.361	None
MV	**0.483**	R2 total, pH, LDH	0.423	None
mMRC	**0.675**	Age, Cr, LDH	0.613	None

2.4.3 Feature rating and target rating by multi-variable analysis

We use the multi-variable analysis model to build the mapping between 53 features and the 16 respiratory sequelae, and the most informative features and most predictable features are listed below:

Supplementary Table 11 The top six feature rating $\boldsymbol{r_{in}}$ and target rating $\boldsymbol{r_{out}}$ during the prediction of respiratory sequelae. The top three most informative features include three radiomics quantified by the DLPE method.

Features	$\boldsymbol{r_{in}}$	**Targets**	$\boldsymbol{r_{out}}$
R2 total	0.105	SGRQ	0.326
LDH	0.080	pO2(A-a)	0.135
Age	0.073	TLC	0.101
R1 total	0.054	mMRC	0.084
pO2	0.051	DLCO/VA	0.071
R1 lower	0.047	ERV	0.069

2.5 Follow-up abnormalities in our cohort

Supplementary Table 12 gives the overview of the respiratory sequelae of COVID-19 on our survivor cohort, with their normal value ranges, the ratio of the abnormal ones in our cohort, and the mean and the worst values in our cohort.

Supplementary Table 12

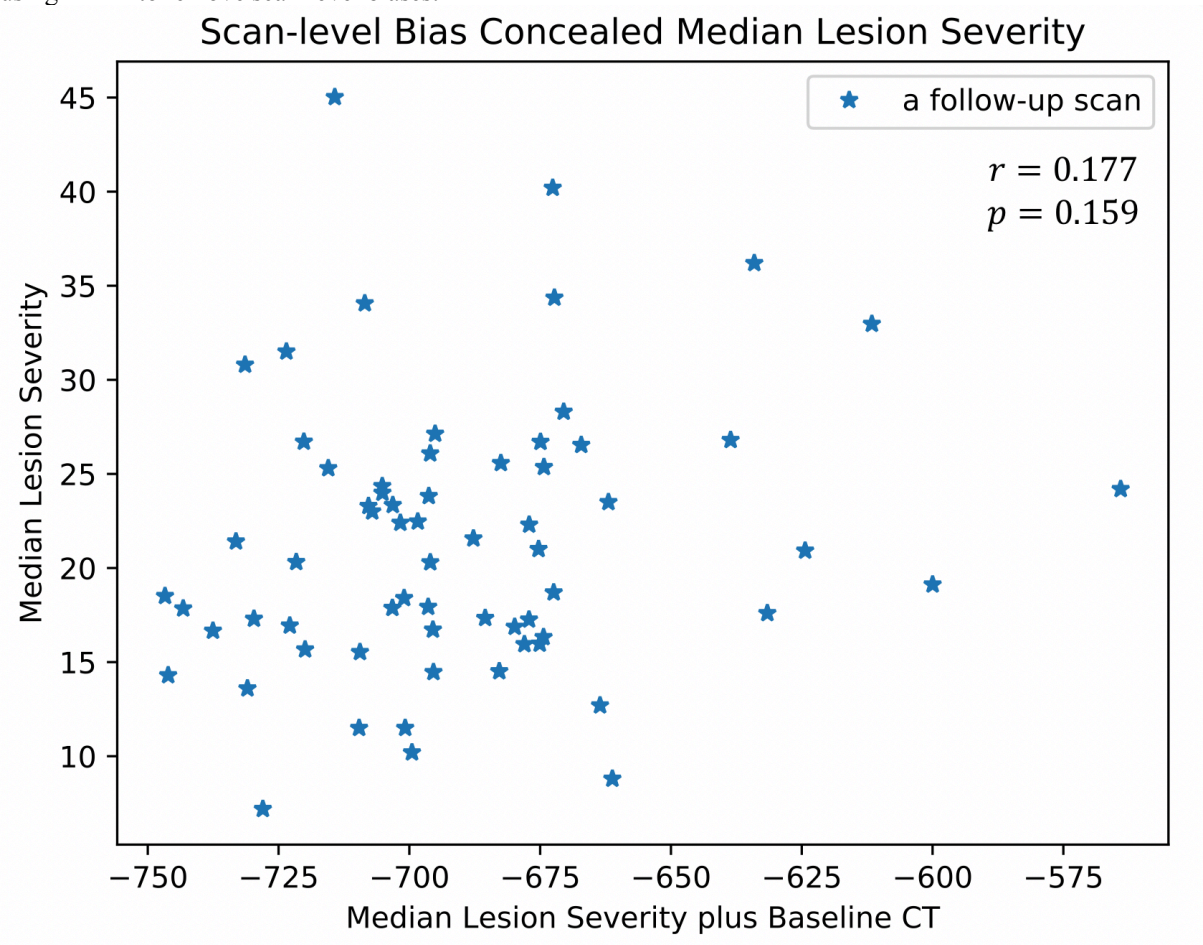
FOLLOW-UP ABNORMALITIES

Sequelae	**Reference Value**	**Abnormal Ratio**	**Mean & Worst**
SGRQ	≤1	43/46	18.6, 49
DLCO	≥80	18/46	83.4, 55.3
MEF50	≥80	17/46	81.2, 31.0
FRC	≥80	17/46	84.9, 58.5
RV	≥80	14/46	88.1, 64.3
FEV1/FVC	≥70	7/46	77.9, 50.7
Chest CT, lung window	No Abnormal CT Patterns	21/46	NA
Chest CT, with DLPE	No Abnormal CT Patterns	38/46	NA

2.6 Other potential applications for the DLPE method

2.6.1 Remove scan-level biases

As shown in the Fig.3 (b) the main text, the scan-level biases can be much larger than the radiomics of the sub-visual lesions here we further give the comparison of the R1 (median lesion severity) with scan-level biases and using DLPE to remove scan-level biases:



Supplementary Fig. 17 Comparison between the median lesion severity before and after removing the bias of the baseline CT. Each data point represents a CT scan in the follow-up cohort, its y-value is the median lesion severity after enhanced (close to the ground truth value), while its x-value is the severity before enhanced, i.e., equals to the median CT signal (HU) for lesion regions on original CT, which contains many biases. The above figure shows that when the major noise (like baseline CT) is not removed by DLPE, the measured features (like median lesion severity) may reflect very few of their ground truth values (here the Pearson correlation coefficient is only 0.177).

2.6.2 Improve segmentation performance and reduce training size

As shown in Fig. 3, using DLPE to remove the biases contributed most to the segmentation of sub-visual lesions. Other lesions like GGO lesions of COVID-19, pulmonary nodules and some lung cancers can also be quite faint. We found using DLPE as a pre-processing methods to remove the biases can help to achieve better performances for the segmentation of pulmonary nodules and COVID-19 lesions, and using DPE to remove biases can achieve the same average dice with less training size. The model, training and testing procedures were exactly the same with our previous TMI work (10.1109/TMI.2020.3001810).

Supplementary Table 13 The segmentation for pulmonary nodules. The best performer is in bold. The performances were shown in the form of average dice. Test set is 55 CT scans from a different hospital of the training set. First row: number of training CT scans. Second row and third row: the performances on the test set when the model is trained on these number of CT scans with and without DLPE enhancement.

Training Scans:	10	25	50	100	179
Dice with DLPE:	0.677	0.826	0.875	0.903	**0.905**
Dice without DLPE:	0.309	0.649	0.813	0.881	0.892

Supplementary Table 14 The segmentation for COVID-19 inpatient lesions (visible lesions). The best performer is in bold. The performances were shown in the form of average dice. Test set is 109 CT scans from a different hospital of the training set. First row: number of training CT scans. Second row and third row: the performances on the test set when the model is trained on these number of CT scans with and without DLPE enhancement.

Training Scans:	10	25	50	100	250
Dice with DLPE:	0.540	0.766	0.811	0.859	**0.899**
Dice without DLPE:	0.364	0.492	0.681	0.818	0.841

Reference

[1] L. Zhou et al., A Rapid, Accurate and Machine-Agnostic Segmentation and Quantification Method for CT-Based COVID-19 Diagnosis. *IEEE Transactions on Medical Imaging* 39(8) 2638-2652 (2020).

[21] RR. Selvaraju et al., Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* 618-626 (2017).

[22] K. Vinogradova et al., Towards Interpretable Semantic Segmentation via Gradient-weighted Class Activation Mapping. *arXiv*:2002.11434 (2020).

[23] K. Wickstom et al., Uncertainty and interpretability in convolutional neural networks for semantic segmentation of colorectal polyps. *Medical Image Analysis* 60, 101619 (2020).