Appendix 1

Model Construction

The Static Model is a combination of MobileNet V2. The MobileNet V2 is used to extract the spatial features, which is shown in *Figure S1*.

MobileNet V2

The structure of MobileNet V2 (22) is detailed in *Table S1*, which had 7 bottleneck stages. All of the bottleneck stages were constructed by a 2-dimensional convolution layer, batch normalization layer, ReLu activation layer, and depth-wise convolution layer (see *Figure S1*).

Mathematics for layers

Convolution

Convolution is a basic operation used to extract the local features in an image (28), which can be expressed as:

$$g(i,j)=\sum_{k,l} f(i+k,j+l)h(k,l)$$

where $f(i,j)$ is the original image, $h(k,l)$ is the convolution kernel, and $g(i,j)$ is the feature image.

Batch normalization

Batch normalization (BN) is a non-parametric operation, which is used to accelerate the training process and address the overfit problem (29). The BN is a practical method to regularize a model and enables higher learning rates. The formulation can be expressed as:

$$\hat{x}=\frac{x-E[x]}{\sqrt{Var[x]+\in}}$$

Where $E[x]$ is the expectation over training mini-batches, and $Var[x]=\frac{m}{m-1}\dot{E}_B\left[\sigma_B^2\right]$ is the unbiased variance estimate over training mini-batches of size m and sample variances σ_B^2. This operation is used to follow the convolution operation.

Non-linear activation

Non-linear activation is a non-linear function that can transform a linear system to a non-linear system. The operation usually uses a map function to enhance the fitting ability of the model. Sigmoid function and ReLU function (30) were used in our models, and both of these operations were element-wise. Sigmoid function is formulated as:

$$f(x)=\frac{1}{1+e^{-x}}$$

and this activation is used in the last dense layer to match the binary cross entropy loss. The ReLU function is expressed as:

$$f(x)=\max(x,0)$$

and this operation is used in each BN operation.

https://dx.doi.org/10.21037/atm-22-3767

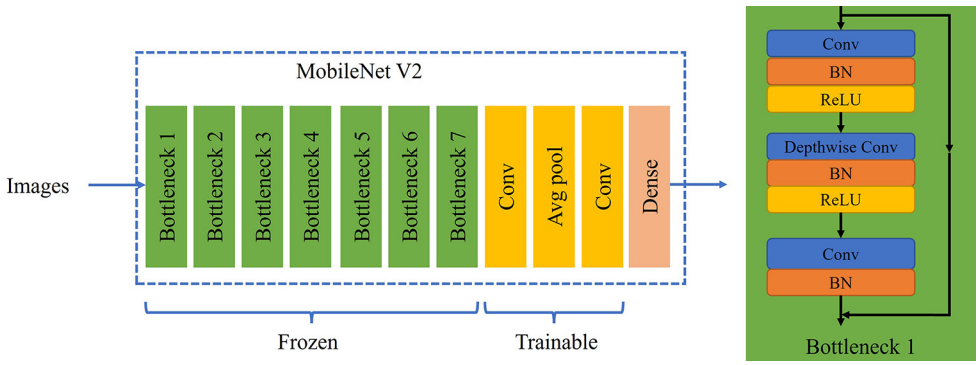

Figure S1 The structure of the static model.

Table S1 Structure of ResNet50

Layers	Output Size	Channel Size	Stride	Expansion
Convolution	112*112	32	2	–
Bottleneck 1	112*112	16	1	1
Bottleneck 2	56*56	24	2	6
Bottleneck 3	28*28	32	2	6
Bottleneck 4	14*14	64	2	6
Bottleneck 5	14*14	96	1	6
Bottleneck 6	7*7	160	2	6
Bottleneck 7	7*7	320	1	6
Conv	7*7	1280	1	–
Average Pool	1*1	–	–	–
Conv	1*1	1	–	–
Dense	1	–	–	–

Pooling

Pooling is a method to reduce the parameters and redundant information. Due to the static property of images, local features can be described by the statistics of local regions. Thus, the operation paid attention to the whole image domain and ignored the local information, which led to the invariance in translation, rotation, and scale. In this study, we use 2-dimensional maximum pooling with strides 2×2 and 2- dimensional global average pooling with strides 2×2 in the first convolution stage and final part.

Constraint

Weight constraint is a way to control the weight that needs to be trained. A weight constraint can keep the updated weight in a small range, which is a strategy for reducing overfit. The ResNet50 is based on ImageNet weights used in our study. Thus, only the last dense layer needed to be set. In this study, the weight constraint was used as the UnitNorm, which can be expressed as:

$$norm(w)=1$$

https://dx.doi.org/10.21037/atm-22-3767

Table S2 The hyper parameters for training process.

Model	LR	BS	Eq	Dropout
Animal Model	1e–07	64	60	0.5
Human Model	1e–07	64	60	0.5

LR, learning rate; BS, batch size; Ep, epoch.

Depth-wise separable convolution

Depth-wise separable convolution is a two-step convolution operation to reduce the parameters of convolution (28), which can be expressed as depth-wise convolution and pointwise convolution.

Depth-wise convolution can be expressed as:

$$g(i.j.c) = \sum_{k,l} f(i+k, j+l, c)h(k,l,c)$$

Where $f(i,j,c)$ is the original image for channel c,$h(k,l,c)$ is the convolution kernel with c filters, and $g(i,j,c)$ is the feature image.

Pointwise convolution can be expressed as:

$$g(i,j,M) = \sum_{k,l,c} f(i+k, j+l, c)h(k,l,M)$$

Where $f(i,j,c)$ is the original image for channel c,$h(k,l,c)$ is the 1*1 convolution kernel with M filters, and $g(i,j,M)$ is the feature image.

Training details

In this study, a clinical model was trained. The model was trained in 2 steps. First, it was trained using animal US images. Second, the model was fine-tuned using a few human US images. For the 1st step, 3-fold CV was used for the animal US images. For the 2nd step, each fold CV model was fine-tuned with the last 3 layers from the same human US images and averaged to obtain the final clinical model. The training process employed an optimization process to obtain the parameters in the deep-learning model. In our study, 4 deep-learning models were trained by gray scale (GS) images. The data were augmented to overcome the overfit in the training process. The augmentation included vertical and horizontal random flipping, random brightness, and random contrast. The classification model sought to predict 2 classes of images. In the training set, binary cross entropy loss was used for the classification, which can be expressed as:

$$L = -\frac{1}{n}\sum_{n}\left[y \ln x + (1-y)\ln(1-x)\right]$$

where the sample number is n,x is the prediction, and y is the ground truth. An algorithm for the 1st-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments (Adam) (31), served as the optimizer. The hyper parameter learning rate (LR), batch size (BS) and epoch (Ep) are listed in *Table S2*.

References

28. Lai S, Xu L, Liu K, et al. Recurrent convolutional neural networks for text classification. Twenty-ninth AAAI conference on artificial intelligence. 2015.
29. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. International conference on machine learning. PMLR, 2015:448-45.
30. Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. Proceedings of the fourteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 2011: 315-323.
31. Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

 https://dx.doi.org/10.21037/atm-22-3767