Supplementary Materials

Predicting microvascular invasion plus cytokeratin 19 expression positivity in hepatocellular carcinoma based on EOB-MRI using multitask deep learning

Yuanyuan Zhao[1,2,#], Xiang Huang[3,#], Meili Sun[4,#], Jia Chen[5], Jian Zhang[6,7], Shiting Feng[8], Jianpeng Li[3], Kangyang Cao[9], Jifei Wang[8], Bingsheng Huang[2], Yujian Zou[3]

[1]Nuclear Medicine Department, The Seventh Affiliated Hospital, Sun Yat-sen University, Shenzhen 518107, Guangdong, China.

[2]Medical AI Lab, School of Biomedical Engineering, Medical School, Shenzhen University, Shenzhen 518055, Guangdong, China.

[3]Department of Radiology, The Tenth Affiliated Hospital of Southern Medical University (Dongguan People's Hospital), Dongguan 523059, Guangdong, China.

[4]Department of Radiology, Sun Yat-sen University Cancer Center, Guangzhou 510060, Guangdong, China.

[5]Active Medical Devices Testing Department, Shenzhen institute for drug control, Shenzhen 518034, Guangdong, China.

[6]Shenzhen-Hong Kong Institute of Brain Science-Shenzhen Fundamental Research Institutions, Shenzhen 518055, Guangdong, China.

[7]Medical School, Shenzhen University, Shenzhen 518060, Guangdong, China.

[8]Department of Radiology, Sun Yat-sen University Affiliated First Hospital, Guangzhou 510080, Guangdong, China.

[9]Faculty of Applied Sciences, Macao Polytechnic University, Macao 999078, China.

[#]These authors contributed equally to this work.

Correspondence to: Prof. Yujian Zou, Department of Radiology, The Tenth Affiliated Hospital of Southern Medical University (Dongguan People's Hospital), No.78 Wan Dao Road, Xingu Yong, Wanjiang Street, Dongguan 523059, Guangdong, China. E-mail: zouyujian@sohu.com; Prof. Bingsheng Huang, Medical AI Lab, School of Biomedical Engineering, Medical School, Shenzhen University, No.1066 Xueyuan Avenue, Taoyuan Street, Nanshan District, Shenzhen 518055, Guangdong, China. E-mail: huangbs@gmail.com; Prof. Jifei Wang, Department of Radiology, Sun Yat-sen University Affiliated First Hospital, No.58 Zhong Shan Er Lu Road, Yuexiu District, Guangzhou 510080, Guangdong, China. E-mail: wjif@mail.sysu.edu.cn

Inclusion and exclusion criteria

1. Inclusion criteria

(1) Patients who underwent Gd-EOB-DTPA-enhanced MRI examination within one month before surgery.

(2) Presented as a single malignant lesion.

(3) No other anti-tumor treatments were administered before surgery.

(4) HCC, confirmed by postoperative pathology; this included clear identification of MVI and CK19 expression.

2. Exclusion criteria

(1) MRI image quality did not meet diagnostic requirements.

(2) Postoperative pathology did not clearly indicate the results of MVI and CK19 expression.

Deep learning classification algorithms

1. ResNet deep learning network

The core idea of ResNet is residual learning, which addresses the issues of gradient vanishing and network degradation caused by increasing network depth. By

introducing residual modules with skip connections, the network is transformed from *H(x) = x to H(x) = x + F(x)*, where *F(x)* represents the residual function to be learned, defined as *F(x) = H(x) – x* (**Figure S1**). This formulation enables the network to learn the residual information, thereby alleviating the training difficulty while mitigating the problem of gradient vanishing associated with increasing network depth. The ResNet50 network used in this study mainly comprises a convolutional layer, a residual block and a fully connected layer.

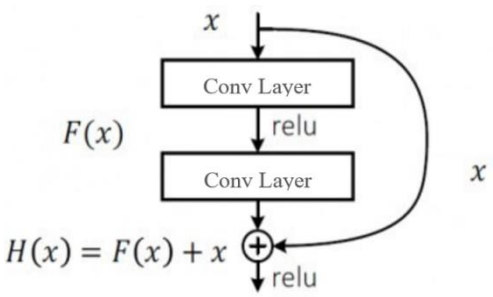

Figure S1 Residual module of ResNet.

2. EfficientNet Deep learning network

EfficientNet is a model scaling method that simultaneously considers speed and accuracy. Unlike previous studies that only make adjustments in one aspect, this method scales the network's depth, width, and input image resolution together. The core module of the EfficientNet network structure is the mobile inverted bottleneck convolution (MBConv), which has a structure similar to depthwise separable convolution and incorporates the compression and excitation operations from SENet. In this study, we utilized the EfficientNet-B0 architecture, which consists of 2 convolutional layers, 16 MBConv modules, 1 global average pooling layer, and 1 fully connected layer. The network employs the Swish activation function, defined as follows:

$f(x) = \frac{x}{1+e^{-\beta x}}$ (β can be a constant or a trainable parameter), （1）

3. ResNeSt deep learning network

The core module of the ResNeSt network is the ResNeSt block containing the Split attention module. As shown in **Figure S2**, the main operation of the ResNeSt block is to split the input feature map k times to obtain k basic blocks and the structure of each basic block. As seen in **Figure S3**, each basic block segments the input feature map and cuts the input feature map into r parts; these r feature maps then enter the distraction area to obtain each slice's attention parameters and output the overall features of r slices. In general, the operation of the ResNeSt block is to split the feature maps input to the module into k×r feature maps and then calculate the attention parameters of these k×r feature maps. Finally, the attention parameters are applied to each feature map, and the final result is output.

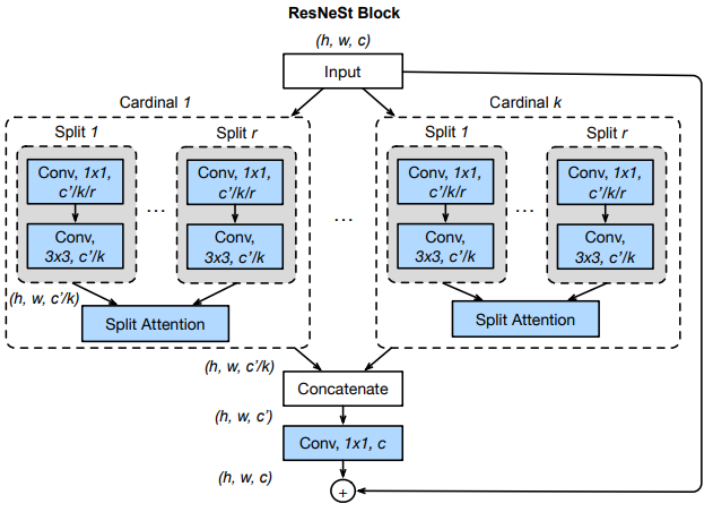

Figure S2 ResNeSt Block.

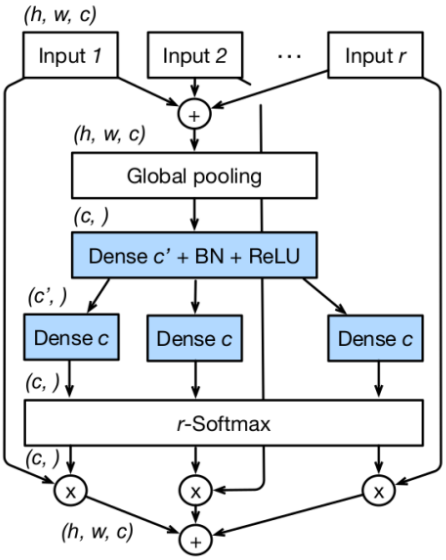

Figure S3 Split attention module.

Multi-task learning framework

1. Hard parameter sharing algorithm

The core idea of the Hard parameter-sharing algorithm is to map the data representations of different tasks into the same latent space. The network model can learn the joint representation between tasks by sharing the hidden layers among all tasks while maintaining separate output layers for specific tasks. The final predictions for different tasks are obtained through the branches corresponding to specific tasks. The hard parameter-sharing algorithm significantly reduces the overfitting risk of the network model. Moreover, the more related tasks are learned, the easier it is for the network model to capture similar features during the learning process. Setting the parameters of the hidden layers to be consistent with most tasks reduces the risk of overfitting individual tasks.

The structure of the hard parameter sharing network model mainly consists of a shared module and specific task modules. The shared module is responsible for

extracting common features for tasks and learning the shared representation between tasks. The specific task modules are designed as network branches for different tasks used to output the final predictions for each task. The basic structure of the hard parameter-sharing network is shown in **Figure S4**.

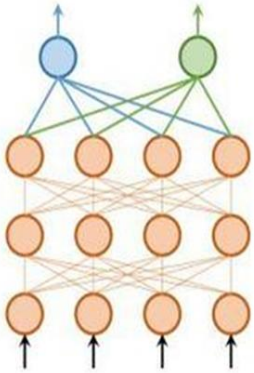

Figure S4. Hard parameter sharing network basic structure.

2. Cross-stitch network algorithm

The core of the Cross-Stitch Network algorithm is the "cross-stitch" structure, which automatically allows the network to learn the ratios of shared features. Essentially, the cross-stitch structure is a coefficient matrix, expressed as shown in Equation 1.

$$\begin{bmatrix} \tilde{x}_A^{ij} \\ \tilde{x}_B^{ij} \end{bmatrix} = \begin{bmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{bmatrix} \begin{bmatrix} x_A^{ij} \\ x_B^{ij} \end{bmatrix} , \qquad (2)$$

where x_A^{ij} and x_B^{ij} are the feature maps output by the Lth layer of two tasks, $\tilde{x}_A^{ij}$ and $\tilde{x}_B^{ij}$ are the feature maps output after the combination transformation, and α is the learnable task weight parameter. α_{AB} or α_{BA} equal to 0 indicates no shared feature between the tasks; when α_{AB} or $\alpha_{BA} > 0$, it indicates a higher ratio of shared features between the tasks. The cross-stitch structure adjusts the degree of sharing

between tasks through the learnable task weight parameters.

The basic structure of the Cross-Stitch Network is shown in **Figure S5**. Almost every layer is designed with a cross-stitch structure. This structure allows the network to automatically adjust the degree of sharing between tasks by changing the task weight parameters. It determines which task's feature parameters need to be shared and at what ratio. Moreover, the cross-stitch structure can freely move between different layers, allowing the sharing mode to be altered by changing its position.

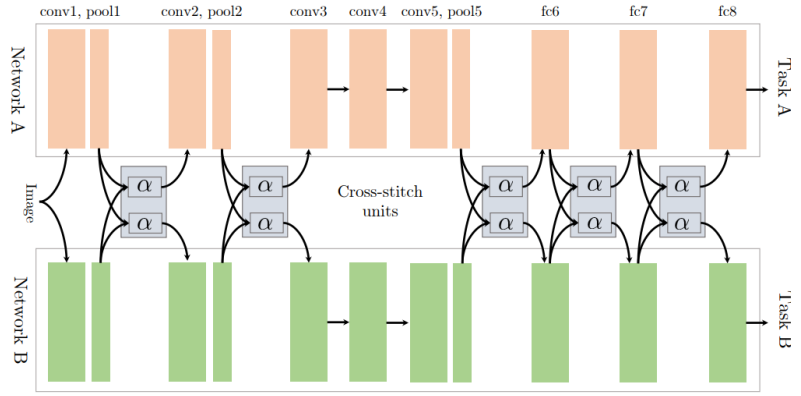

Figure S5. Cross-stitch network basic structure.

3. Experts sharing network

Currently, most multi-task learning algorithms face two challenges: negative transfer and the inability to improve the performance of multiple tasks simultaneously. It is common to observe a situation where one task's performance improves while another's performance deteriorates.

Figure S6a shows the Hard parameter sharing network model. The network structure shows that the CK19 expression prediction and the MVI prediction tasks share a feature extraction module. However, this may lead to negative transfer phenomena due to weak correlation or conflicts between tasks. Additionally, one task may

dominate the feature-sharing module of the network model, increasing performance for that task, while another task's performance may decline. **Figure S6b** shows the Cross-Stitch Network model in multi-task learning. This model automatically adjusts the degree of sharing between tasks by changing the task weight parameters in the cross-stitch module. The network autonomously learns linear combination weights based on the tasks to integrate representations from different tasks selectively. This approach partially addresses the issue of negative transfer in the network. However, there may still be cases where one task's performance improves while another task's performance declines. To address these issues, we propose a network structure that separates shared tasks and specific task parameters (**Figure S6c**), to avoid conflicts caused by complex parameter sharing and to mitigate the associated instability in the network model.

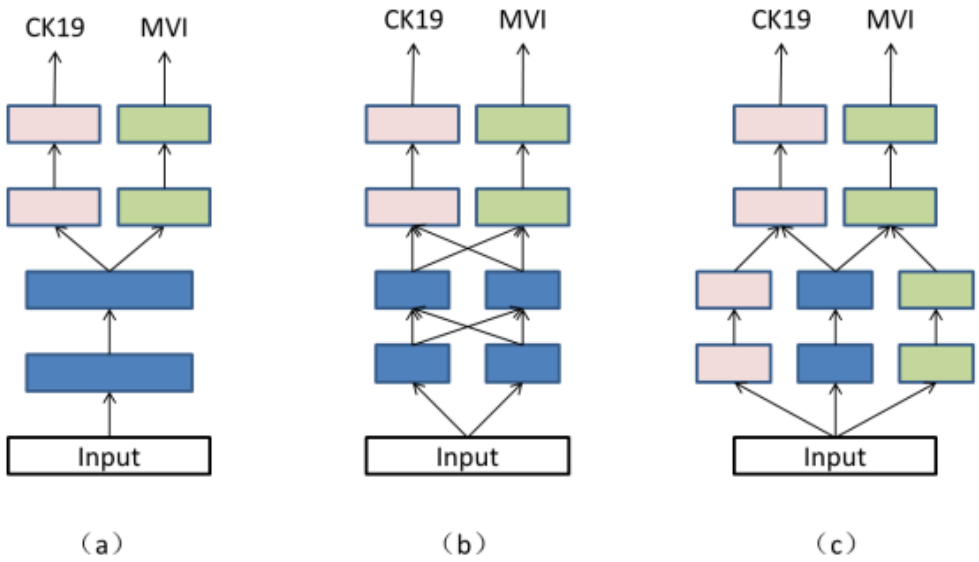

Figure S6. Basic structures of multi-task learning networks. (a) The hard parameter sharing network (b) The cross-stitch network. (c) The expert-sharing network.

The core idea of the Expert-Sharing Network is to separate shared and individual task parameters, thereby avoiding the instability caused by complex parameter sharing in the network model. The basic structure is illustrated in **Figure S6c**. The feature

extraction part of the network is divided into three sub-networks: Sub-Network 1, which learns image features specifically for CK19 expression prediction; Sub-Network 2, which learns image features specifically for MVI prediction; Expert-Sharing Sub-Network, which simultaneously learns common image features for both CK19 expression and MVI prediction tasks, facilitating shared learning. After the feature extraction learning through the three sub-networks, the branch network of the CK19 expression prediction task incorporates the feature knowledge from Sub-Network 1 and the Expert-Sharing Network, while the branch network of the MVI prediction task incorporates the feature knowledge from Sub-Network 2 and the Expert-Sharing Network. Each branch network then outputs the prediction results for the respective tasks.

From the structure of the Expert-Sharing Network, it can be observed that each sub-network is influenced only by its respective task during the learning process, while all tasks influence the Expert-Sharing Sub-Network. This structure enables learning both task-specific feature knowledge and shared knowledge between tasks. It helps avoid network bias towards fitting a particular task due to complex parameter sharing, thereby preventing negative transfer and situations where one task's performance improves while another task's performance declines.

The network structure of improvement

1. Spatial Transformation Module

The basic structure of the spatial transformation module is shown in **Figure S7**. The spatial transformation module performs affine transformation operations on the feature map and is mainly divided into three parts: (1) the localization network, which generates the affine transformation coefficient matrix θ; (2) the grid generator, which generates the sampling grid for the final output feature map. It begins by generating a

grid based on the size of the final output feature map and then applies the affine transformation coefficient matrix θ generated by the localization network to obtain the sampling grid corresponding to the output feature map. (3) The sampler, which performs position mapping on the input feature map to obtain the output feature map after affine transformation. The module uses interpolation to fill in the missing values in the regions without values.

In this study, the spatial transformation module we designed has the localization network as its core component, which consists of 3 convolutional layers and 2 fully connected layers. The convolutional layers have kernel sizes of 1x1, 3x3, and 5x5, respectively, and are followed by a leaky ReLU layer.

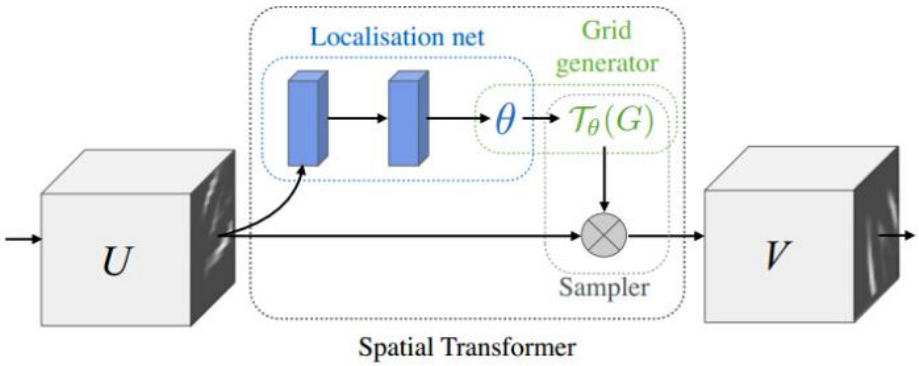

Figure S7. The basic structure of the spatial transformation module.

2. Relation Reasoning Module

The relation reasoning module aims to leverage the mutual relationships between tasks to enable more accurate judgment and prediction by the model. The relation reasoning module we designed analyzes the potential relationships between tasks in a data-driven manner without the need for prior knowledge about which relationships exist. The relation reasoning module in our study can be represented by a simple composite function, as shown in Equation 3:

$$\mathrm{LR}(A_l, A_m) = f_{\emptyset}(g_{\theta_{lm}}(A_l, A_m)), \qquad (3)$$

where the inputs A_l and A_m in the equation represent the feature maps extracted from a pair of tasks for which the correlation needs to be computed; the functions $g_{\theta_{lm}}$ and $f_{\emptyset}$ are responsible for capturing the paired relationships between tasks, i.e., inferring whether there exists a relationship between two objects and how they are correlated. The basic structure of the relation reasoning module is depicted in **Figure S8**.

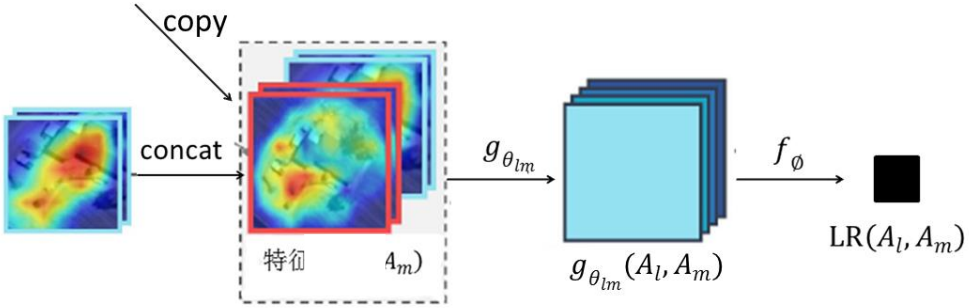

Figure S8. The basic structure of the relation reasoning module.