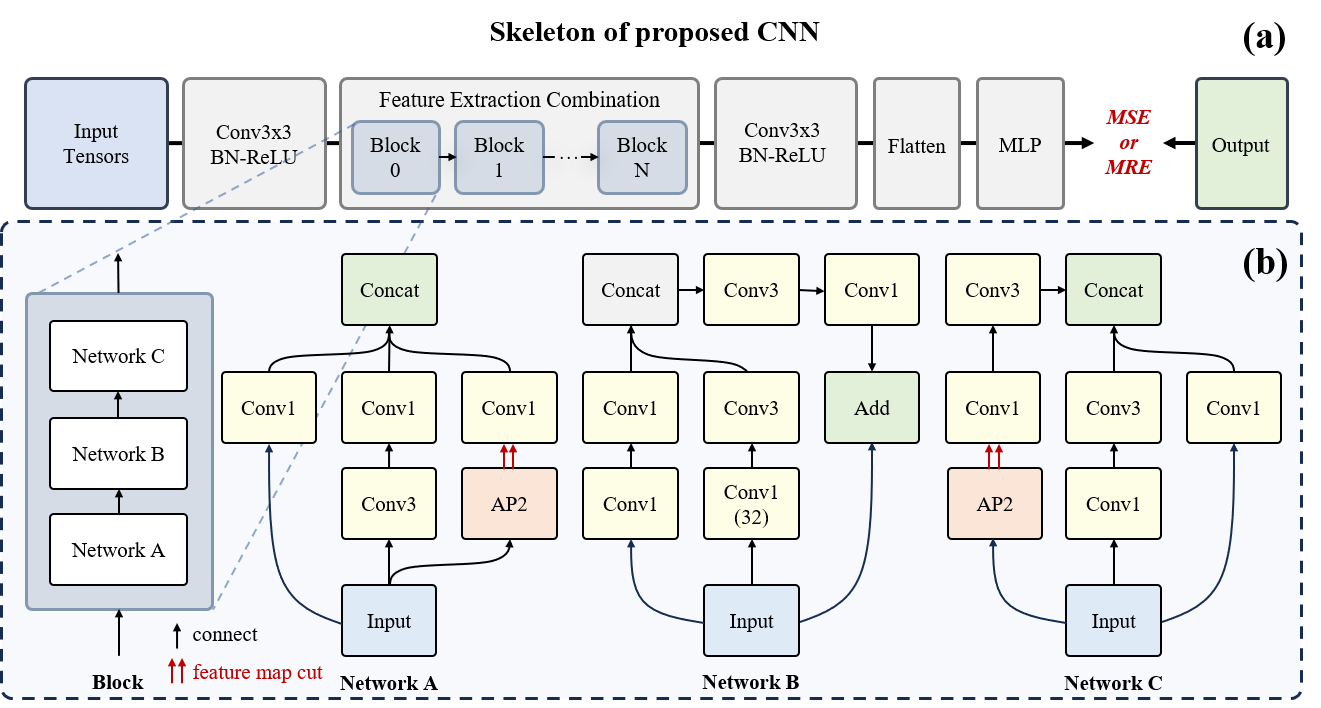
**S2 Details of Subnetworks in Each Block**

Figure 1 (b) in the main article demonstrates the internal structure of each block, each of which is stacked by three sub-networks, namely Network A, Network B, and Network C. Please note that in the bottomhalf figure, “Conv x” refers to a group of operations that sequentially connect: a convolutional operation with kernel size x, a batch normalization layer, and the activation function ReLU. “AP2” refers to an average pooling operation with kernel size two and the padding is set to one. Please note that the “AP2” operations are always connected by a feature map cut operation (shown in red in Figure 4), which clips the 16×16 feature map to 15×15, keeping the feature map size invariant. The numbers with brackets in each box denote the output channels of each module. The detailed sub-networks architectures and their design motivations are as follows:



**Fig. 1**: The network structure used in this work. (a): The skeleton of the network

structure. (b): The detailed structure of extraction block.

– Network A: In the initial stage, a 32-channel input is subjected to a 3×3 convolutional layer (Conv3), resulting in a 64-channel feature map. Subsequently, an 1 × 1 convolutional layer (Conv1) reduces the channels to 32. An average pooling layer (AP2) is then employed for dimensionality reduction. Another 1×1 convolutional layer further decreases the channels to 32. The final structure comprises a repetition of a 64-channel 3×3 convolutional layer and a 32-channel 1×1 convolutional layer. Network A encompasses lo- cal and global feature extraction through the combination of 3 ×3 and 1×1 convolutions. Additionally, 1 × 1 convolutions aid in controlling the model’s parameters by reducing the depth of feature maps, and the average pooling layer contributes to dimensionality reduction, potentially mitigating computational load and preventing overfitting.

– Network B: Commencing with an 128-channel input, Network B under- goes a series of convolutional operations. A 64-channel 1×1 convolutional layer (Conv1) is followed by a 32-channel 1×1 convolutional layer. After a 64-channel 3×3 convolutional layer, an “Add” operation combines the preceding layers with the output of a 128-channel average pooling layer (AP2). This is achieved by a sequence of a 32-channel 1×1 convolutional layer, a 64- channel 1×1 convolutional layer, and a 32-channel 3×3 convolutional layer. The final step involves another “Add” operation, merging the output of the current sequence with the output of the previous layers. Network B encompasses feature fusion through “Add” operations, potentially leveraging residual learning [for gradient vanishing issues while maintaining information](#bookmark40) flow. The mix of 1×1 and 3×3 convolutional layers aids in information retention and parameter efficiency, achieved by controlling model parameters through changes in convolutional layer channel counts.

– Network C: Similar to Network B, Network C initiates with an 128-channel input and proceeds through a 64-channel 1×1 convolutional layer, followed by a 32-channel 3×3 convolutional layer. Immediately succeeding is an 128- channel average pooling layer (AP2). A feature fusion “Add” operation combines the output of the average pooling layer with that of the previous 3 ×3 convolutional layer. This is followed by a 16-channel 3×3 convolutional layer, a 64-channel 1×1 convolutional layer, an 8-channel 3×3 convolutional layer,and a 32-channel 1×1 convolutional layer. Network C encompasses deep feature integration achieved through the alternating use of different-sized convolutional kernels and pooling layers. Similar to Network B, the combination of average pooling and “Add” operations aids in dimensionality reduction while retaining important information. The network is designed to progressively refine and strengthen feature representations by transitioning from larger to smaller channel numbers.

In general, the design of these three sub-networks draws inspiration from multi-scale processing, as evidenced by the inclusion of convolutional and pooling layers of different sizes, indicating a consideration for handling features at various scales. This is crucial for understanding more complex visual patterns and has been widely studied. We also comply with the basis of achieving feature reuse and fusion: The final concatenate operation ensures that features learned in all branches are preserved and utilized. Such fusion contributes to enhancing the capability of feature representation, thereby improving the accuracy of final decisions.