The implementation of neural network computation flow

a) Fully connected layer: Suppose an input matrix $\mathbf{H}^l$ to a fully connected layer is of size $B \times D_{\text{in}}$, l is the index of the layer. B is the batch size, which is supposed to be a multiple of ensemble size K. For fully connected layer, a weight matrix $\bar{\mathbf{\Theta}} \in \mathbb{R}^{K\times D_{\text{out}}\times D_{\text{in}}}$ contains K weight matrices from different members. The input is first split into $[\mathbf{H}_1^{'l}, \dots, \mathbf{H}_K^{'l}]$. Each $\mathbf{H}_k^{'l}$ is of size $B' \times D_{\text{in}}$, where $B' = \frac{B}{K}$. The output matrix is $\mathbf{H}^{l+1} \in \mathbb{R}^{B\times D_{\text{out}}}$, composed by $[\mathbf{H}_k^{'l+1}]_k$, where $\mathbf{H}_k^{'l+1} \in \mathbb{R}^{B'\times D_{\text{out}}}$, $\mathbf{H}_k^{'l+1} = \mathbf{H}_k^{'l}\bar{\mathbf{\Theta}}^{kT}$. The computation is efficiently implemented with Einstein summation convention in modern computation library [1], [33].

b) Convolutional layer: For convolutional layer, the input tensor is $\mathbf{H}^l \in \mathbb{R}^{B\times D_{\text{in}}\times\kappa\times\kappa}$, where D_{in} is the number of output channels, D_{in} is the number of input channels, κ is the kernel size and B is the batch size. The weight matrix is $\mathbf{\Theta} \in \mathbb{R}^{KD_{\text{out}}\times D_{\text{in}}\times\kappa\times\kappa}$. The group convolution [24] could be adopted for efficient computation without splitting the input. For the group convolution to work on channels in different ensemble members, the following permutation of the input matrix should be executed.

$$\begin{aligned} \mathbf{H}^l \in& \mathbb{R}^{B\times D_{\text{in}}\times\kappa\times\kappa} \\ \rightarrow \mathbf{H}^{'l} \in& \mathbb{R}^{\frac{B}{K}\times(KD_{\text{in}})\times\kappa\times\kappa} \end{aligned} \tag{21}$$

The group convolution is then executed on $\mathbf{H}^{l+1} = \text{group\_conv}(\mathbf{H}'^{l}, \bar{\mathbf{\Theta}})$. Specifically, the weights indexed by $kD_{\text{in}} \sim (k+1)D_{\text{in}}$ will only be fed to the convolutional computation with $\bar{\mathbf{\Theta}}^k$. Note that the permutation in Eq. 21 will only be executed once, at the input side out neural network.

The batch normalization in the convolutional layer follows the same idea of using group [41]. As the ensemble dimension is already folded in the channels, group normalization could directly work on the layer activations with interfering different members.