Documentation of Kernel Pruning

In kernel pruning, we turn some kernels of convolution layers to zero so that they decrease size of model as well as decrease the complexity of model.

Working:-

1. Get the model and percentage of pruning you want to apply
2. Calculate the number of kernels and total number of convolutional layers

```
total = 0
total_kernel=0
for m in model.modules():
    if isinstance(m, nn.Conv2d):
        total += m.weight.data.numel()
        oc,ic,h,w=m.weight.size()
        total_kernel+=m.weight.data.numel()/(w*h)
```

3. Get the original weights of convolutional layers

```
conv_weights = torch.zeros(total)
conv_max_weights = torch.zeros(total)
index = 0
for m in model.modules():
    if isinstance(m, nn.Conv2d):
        size = m.weight.data.numel()
        conv_weights[index:(index+size)] = m.weight.data.view(-1).abs().clone()
        oc,ic,h,w=m.weight.size()
        weight_max=torch.max(m.weight.data.abs().view(oc,ic,w*h),-1)[0].view(oc,ic,1,1).expand(oc,ic,h,w)
        conv_max_weights[index:(index+size)] = weight_max.contiguous().view(-1).clone()
        index += size
```

4. Then calculate the threshold_index by multiplying total number of convolutional layers with percentage of pruning we want to do.
5. Then sort the convolutional weights and then index on it with threshold_index to get the threshold weight
6. Then creating a masking tensor by going to each weight and checking if the weight is greater than the threshold.

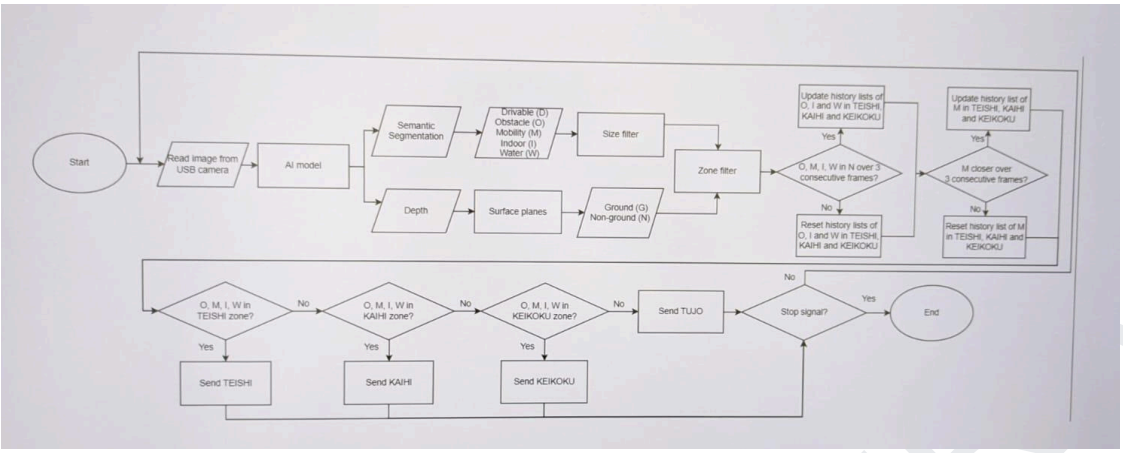
7. Then simply use the masking tensor on original weights

```
for k, m in enumerate(model.modules()):
    if isinstance(m, nn.Conv2d):
        size = m.weight.data.numel()
        oc,ic,h,w=m.weight.size()
        mask = conv_max_weights[index:(index+size)].gt(thre).float().detach().view(oc,ic,h,w)

        pruned = pruned + mask.numel() - torch.sum(mask)
        m.weight.data.mul_(mask)
        index += size
        if int(torch.sum(mask)) == 0:
            zero_flag = True
```

8. Pruning is done

In reference to the flow of overall program, it comes after AI model as pruning is being done after making AI model.



Research paper : Structured Pruning of Deep Convolutional Neural Networks

Link :
https://arxiv.org/ftp/arxiv/papers/1512/1512.08571.pdf#:~:text=The%20kernel%20level%20pruning%20is,researches%20%5B7%5D%2D%5B12%5D.

Various ways of Pruning a convolution layers in ml models

Taking computational advantages using randomly scattered unstructured sparsity in a network is very difficult. It demands many conditional operations and extra representation to denote the location of zero or non-zero parameters. Generally, the convolution layers in the non-pruned network have fully connected convolution connections. In Fig. 2, the layer L_1, L_2 and L_3 contain 2, 3, and 3 feature maps, respectively. As channel and feature map are similar concepts, we therefore used them interchangeably throughout this article. The number of convolution connections between L_1 and L_2 is 6 (=2×3) and that between the L_2 and L_3 are 9. Each feature map in L_2 has a $K \times K$ convolution connection from each channel in L_1. Thus, the

1 pruning exploiting the largest granularity is deleting a feature map or feature maps. If a feature map in a layer is removed, all the incoming and outgoing kernels are pruned. Fig. 2 shows 5 pruned kernels with a red dashed line. Considering the configuration in Fig. 2, the 2-3-3 architecture is reduced to 2-2-3.

2 The next level pruning is deleting kernels where each kernel represents one whole convolution. The kernel level sparsity is depicted with blue dotted lines in Fig.2.

3 The lowest level pruning is using the intra-kernel sparsity, which forces some weights into zero valued ones. In the previous works, the intra-kernel level pruning is usually conducted by zeroing small valued weights [7] [8]. We particularly explore the intra-kernel level pruning using the sparsity at well-defined locations, which is called the intra kernel strided sparsity.

Out of all these three methods we have used Intra kernel pruning in which several small valued weights are turned to zero. For which the procedure is mentioned on first pade of document