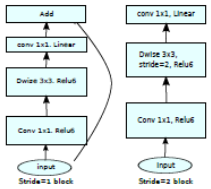
Andrew ID: cmysorej

Below are the details of each CNN later used in my implementation:

Input Channel	Output Channel	Expansion Factor	Iterations (N)	strides(As mentioned in MobileNetV2 paper)	strides(Actual Used)
3	32	1	1	1	1
32	16	1	1	1	1
16	24	6	2	2	1
24	32	6	3	2	1
32	64	6	4	2	1
64	96	6	3	1	1
96	160	6	3	2	2
160	320	6	1	1	1
320	1280	1	1	1	1

1. As it can be observed, strides differ from the strides mentioned in paper. Intuition behind this change was the fact that our image size in 32X32 whereas images used in Mobilenetv2 paper is 224X224 image. So, to avoid reduction of features in each layer, reduced the strides.
2. Shortcuts for Stride 1 layers Not added as mentioned in below diagram of the paper. Initial I tried with both shortcut and without shortcut and the output did not change much. So, ended up without shortcut.
   Reasoning behind Not using shortcut: Now I have lot many layers with stride 1. If I add short cut for every layer with stride 1, I will be giving direct path between input and output of model. This might lead to overfitting for the given input. I guess this helped me with verification task.



(d) Mobilenet V2

3. **Loss used** for both classification and verification: Cross Entropy Loss
4. In the experiments section(section 6) of the mobilenetv2 paper, it is mentioned that they used RMSprop, but upon initial experimentation **SGD performed better** with momentum of 0.9. So used SGD optimizer with **momentum of 0.9 and weight decay of 5e-5**
5. **Training details: (used same for both classification and verification)**
   Batch size: 128 (wanted to use 256 to make training faster but was getting cuda memory error. So ended up with 128)
   No. of epochs: 17
   Learning rate: for epochs 1-15 : 3e-2
   Learning rates: for epoch 16 and 17 : 0.1*3e-2
   I could have done with lesser epochs, but I realized I haven't added learning rate decay (as mentioned in section 6 of the paper) only at 15 epoch.
   I did not worry much of overfitting because the training data and testing data was same for classification.
6. **For verification,** I used the same parameters learnt in classification section used medium set of data. But when I got the output of forward function of model, I normalized the output vector
7. **For verification,** I used cosine similarity as mention in handout. It was clearly giving better results than L2 distance.
8. **Final results:**

Task	Kaggle score
Classification	0.78115
Verfication	0.94159