Deep Residual Learning for Image Recognition

Summary of the paper –

The deeper neural networks use to have a vanishing gradient problem. The vanishing gradient problem is a challenge encountered during the training of deep neural networks, where the gradients used to update the model's weights become extremely small as they are propagated backward from the output layer to the earlier hidden layers. This problem was addressed using normalization layers which enabled networks with tens of layers to start converging for stochastic gradient descent with back propagation. But when the deeper networks started converging a degradation problem was exposed; with the network depth increasing the accuracy saturates and then degrades. This paper addresses the degradation problem by introducing a deep residual learning framework. This framework suggests a simple change: instead of asking a block of layers to relearn the entire output, H(x), we ask it to learn only the residual mapping or the difference between the input and the perfect output, F(x). This difference is what needs to be fixed. The final result is then achieved by taking the original input, x, and simply adding that learned difference, F(x), to it. This addition happens through a skip connection, which is just a shortcut around the main layers. This shortcut makes the network much easier to train because if a layer doesn't need to change the data, the learning process can easily set the layer's weights to zero. The data then flows perfectly through the shortcut, ensuring that training doesn't get messed up, allowing us to build networks with hundreds of layers.

Resnet explanation - https://www.youtube.com/watch?v=woEs7UCaITo

My approach – I started the project by creating a ResNet-50 model using transfer learning in Tensorflow and then I did the same with PyTorch. In both of these of approaches I could not run many epochs and I had to reduce the number of images as well for faster training of the model, but still the accuracy for both of these models was above 70%. And, then I created the ResNet-18 model from scratch using PyTorch. Again, I could not run many epochs and had to reduce the number of images too. The accuracy was too low for this model (around 20%). Then I provided the GUI for the project. I used streamlit for creating the GUI and I selected the pytorch transfer learning model as it gave me the highest accuracy.