DLCV - Hackathon

Private Leaderboard - #1

Naman Jaswani (namanjaswani@iisc.ac.in)
Pratik Likhar (pratiklikhar@iisc.ac.in)

Introduction:

This hackathon was a part of DLCV-2021 course curriculum, in which we were asked to compare two images (one before operation and the other post operation) and classify them into two classes (same patient {1} or different {0}).
Train dataset consisted of 1000 train image-pairs while Test dataset consisted of 5000 image-pairs to predict on.

Dataset Formulation:

We created class specific image-pairs for training, i.e for label {1} we took true image-pairs from /Post and /Pre directories, and for label {0} we kept the image from /Post directory as such, while randomly picked image from /Pre directory (excluding true image from /Pre folder)

eg:

Post_Image	Pre_Image	Label
Post_img_1.jpg	Pre_img_1.jpg	1
Post_img_1.jpg	Pre_img_97.jpg	0

Approach:

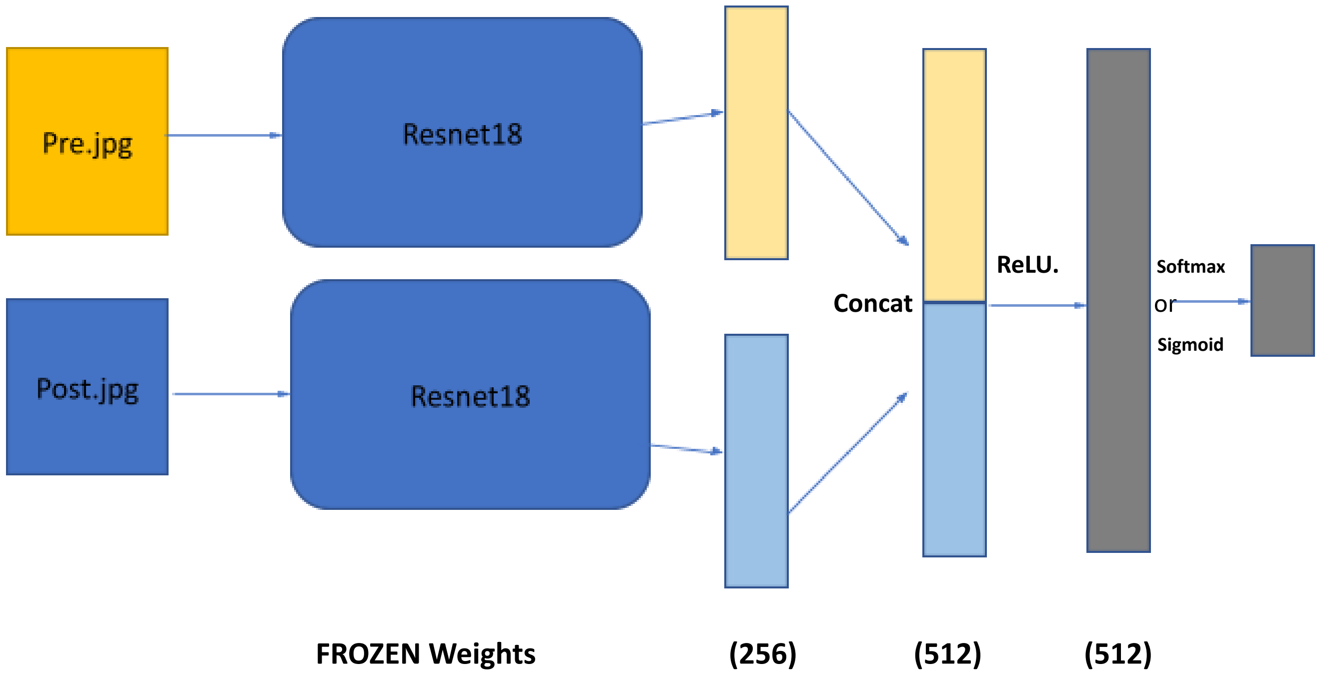
Naman's approach:

1. A two-channel approach was used in which both images (pre_img and post_img) were fed into two separate pretrained models (resnet18) with all weights frozen except for the final fully-connected layers.

2. The feature vectors (256x1) from the two resnet models were then concatenated (512x1) and fed into a feed-forward neural network.

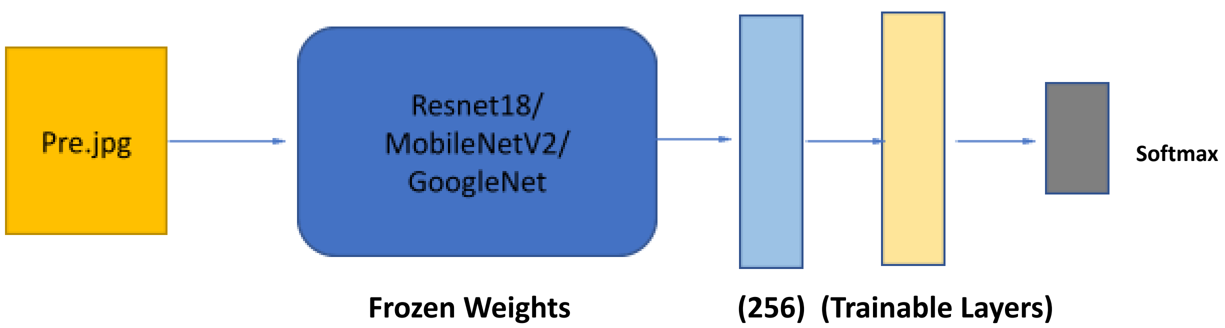
3. Trainable layers : All fc layers only.

4. This architecture was trained in 3 separate ways:
    a) Softmax activation with CrossEntropy loss
    b) Sigmoid activation with CrossEntropy loss
    c) Softmax activation with cropped input images to focus on face only.



Pratik's approach:

1. A single channel approach was initially used in which the two images{pre and post op} were concatenated and fed into a single pretrained model(different architectures were experimented with like Resnet18, MobilenetV2 and GoogleNet) with all weights frozen except for the final fully-connected layers. The loss function for all the models was chosen to be the crossentropy loss.



2. The above approach resulted in a forever decreasing training loss but first decreasing and then increasing validation loss. Tried to tune the hyperparameters, got somewhat stable training but the test score was not good.

3. Then shifted to dual-channel architectures(same as Naman's with softmax activation at the final layer) and experimented with GoogleNet, MobileNetV3Small, Resnet18. Found out that MobileNetV3Small as a feature extractor is giving stable training with high training and validation F1 scores.

4. Finally selected two best models which were:
   a) MobileNetV3Small as the feature extractor without learning rate scheduling.
   b) MobileNetV3Small as the feature extractor utilizing Multi-Step learning rate scheduler.

Final Submission:

Our final submission was a **weighted ensemble** of our all 5 models, where weights were given manually corresponding to each individual model's public LB score.

Things we could have tried:

These are the methods we could have tried, but couldn't due to time constraints:
1. Label smoothing
2. Pseudo Labelling
3. Better architectures (EfficientNet etc.)
4. Learning the ensemble instead of using manually selected weights.

End Note:

The hackathon experience was really fun, as we implemented our own ideas all by ourselves and with great collaboration spirit, we finally won !