Deep Learning(COSC 2779) – Assignment 1

Yogesh Haresh Bojja (s3789918)

1. Problem Definition:

In this assignment we have to predict human actions and classify it to the respective 'action_class' from RGB images using Deep Learning.

2. Approach of modelling:

We will be doing transfer learning and then training our model. We will check and try to reduce the generalization gap of the model by using different techniques like augmentation, regularization, dropout, selecting appropriate activation functions and optimizers. Later on, this tuned model will be fine-tuned again by unfreezing the top layers of the base model and training it again as top layers extract most important features.

At last, we will be testing this newly generated model on the unseen data from S40AR_test_data.csv and store the predictions in s3789918_predictions.csv.

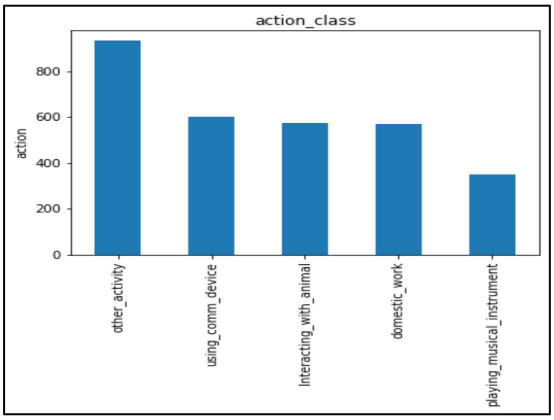
3. Evaluation plan of the model:

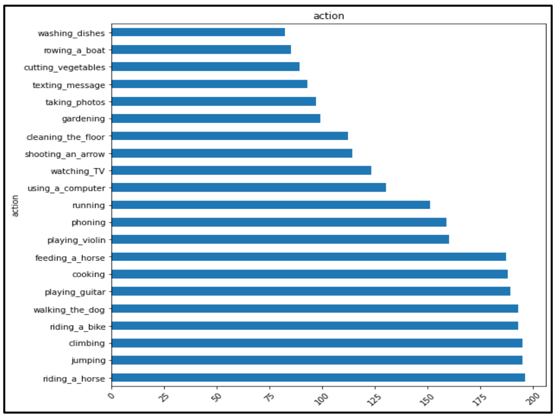
For evaluating our model, we have used 'Categorical cross-entropy' and 'Accuracy' metrics. Our data has two outputs i.e., action and action_class, and each output have multiple classes. In such case it is better to use sparse categorical cross-entropy or categorical cross-entropy. Sparse categorical cross-entropy is used when target column is integer encoded and categorical cross-entropy is used when target column is one-hot encoded. In our case we have one-hot encoded the target class hence we will be using categorical cross-entropy for evaluating the loss. Along with categorical cross-entropy we have considered accuracy as well for evaluating the models.

We have split the data from S40AR_train_data.csv into training (80%), validation (10%) and test (10%) sets.

4. Detailed(step-by-step) explanation of the modelling:

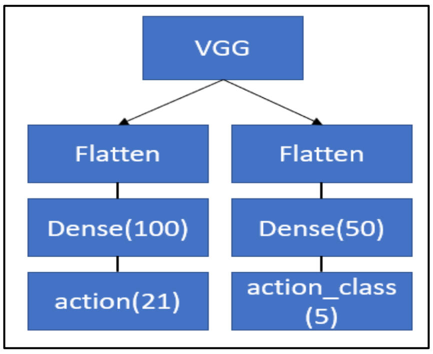
- **Data loading and splitting:** We have split the data into 80% training, 10% validation and 10% testing sets. Thus, there are 2424 images for training and 303 images respectively for validation and testing.
- **Data Exploration:** In this phase we have checked how the data is and what are its dimensions. There were RGB images, one image in the dataset had one channel so while testing its dimensions needed to be expanded. Images in the dataset were of different sizes. Following chart shows the distribution of the classes for output action and action_class.





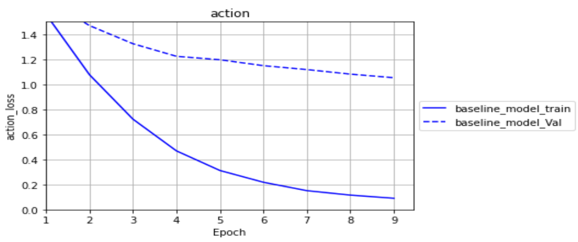
‘Action_class’ seems to be equally distributed while ‘action’ is not equally distributed. This can cause bias in the model.

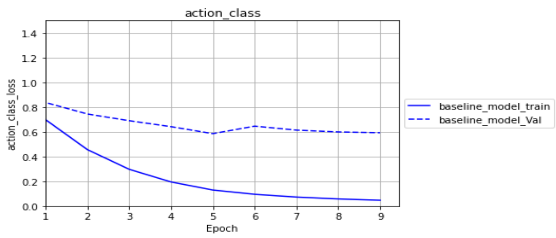
- **Selecting base model:** We will be doing transfer learning so that our model can be trained from the pretrained model which eventually will save time and resources for us. For this assignment, I have selected VGG16 as the base model. Output of this base model is branched and connected to the dense layers to get action and action_class as the output.



Output of VGG16 is given to two branches and then is flattened. We could have created one flattened layer and then give its output to two branches, but we selected to create flatten vectors separately for action and action_class as its backpropagation weights won’t interfere with each other. As there are 21 actions we have selected dense layer with 100 units for predicting action whereas for action_class with 5 classes the dense layer is selected with 50 units.

- **Train baseline model:** We trained the model without augmentation, and we found out that generalization gap is large. To reduce the generalization gap, we will generate different sample of the data via augmentation.
  Categorical cross-entropy of baseline model is shown below –

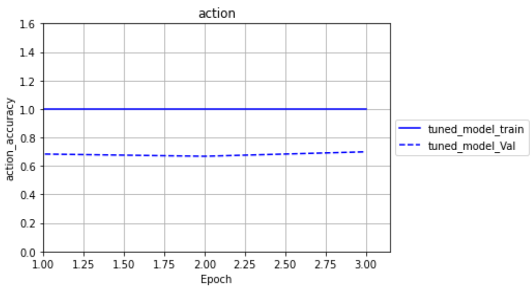


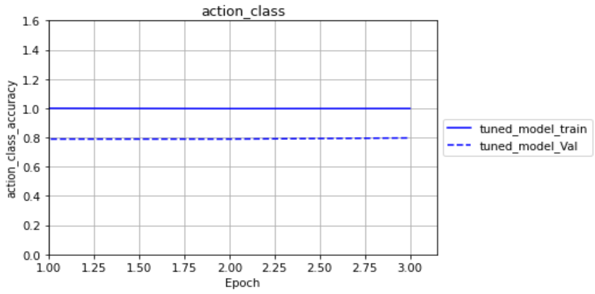


- **Train model with augmentation:** Augmentation had minimal improvement in reducing the generalization gap, but the improvement was not that great.
- **Selecting activation function:** We trained the model on different activation functions and among all those ‘elu’ seemed good for action output and ‘leaky_relu’ seemed good for predicting action_class.
- **Selecting regularization:** We trained the model with L2 and L1 regularization keeping learning rate equal to 0.001. L2 regularization performed well that the L1 regularization. Hence we will select it.
- **Selecting dropout:** Dropout is selected between 0.2 to 0.5. Therefore, we checked it for 0.3 and 0.4 and we found that 0.4 performed well. Thus, along with L2 regularization we will select 0.3 as the dropout value.
- **Check dropout and regularization combined**: When we train our model with dropout=0.3 and L2 regularization there is no significant change in the error loss. We will keep both of these values to train the model as dropout will reduce the computation cost and be beneficial for the model.
- **Selecting optimizer**: Model is trained on different optimizers like SGD, RMSprop, Adam. Adam seems less overfitting in predicting both action and action_class. Thus, we will select Adam as the optimizer.

- **Saving model**: Model trained with dropout=0.3, L2 regularization, and Adam optimizer is saved and loaded further for fine tuning of the top layers of VGG16.
- **Fine tuning of tuning of VGG16**: As we Know that top layers are most important layers to detect features, we will train our top layers of VGG16 by unfreezing it. We will be training top 2 layers. After this we can see that accuracy for action output increased slightly and for action_class it remained same.

  Accuracy of tuned model is shown below –





- **Prediction**: Lastly we have predicted the unseen data from S40AR_test_data.csv and stored our predictions in s3789918_predictions.csv.

5. Justification:

- **Why transfer learning:** Instead of building the model from scratch and training it for huge epochs and for more time, transfer learning does it quickly.
- **Number of Epochs:** As we are using Transfer learning, number of epochs used in the assignment are less i.e., 8 or 10 epochs for each analysis. Due to transfer learning, we were getting high training accuracy in 2nd or 3rd epoch itself thus there was no need of training for huge epochs like 250 or 300.
- **Why VGG16?:** VGG16 is trained on IMAGENET database and has good accuracy. Instead of having huge number of hyper-parameters it is giving good accuracy using CNN model.
- **Why 2 flatten layers in the model?:** We have given output of VGG16 to two flatten layer, instead we could have given it to one flatten layer and then branch it, but we used former approach as back-propagation weights would be different for each branch.
- **Bias initializer:** We have initialized the weights with zeroes instead of random values for better performance.

6. Ultimate judgement: I would recommend using the tuned model we created after applying dropout, appropriate regularization, optimization, augmentation and fine-tuning top layers of VGG16. This tuned model gives slightly better performance than the baseline model.

7. Limitation:

Data given to us is small and after train_test split it shortens further. Classes of ‘action’ is not equally distributed thus generating bias towards maximum occurring classes. To make classes of action equally distributed we cannot delete the data.