Neural Networks Project Report

Kaustubh Butte,Aman Agarwal,Suyash Agarwal

The following text is a writeup on how we reached the final model of our Neural Networks course project. The project problem involved visual question answering on CLEVR dataset. The images and a corresponding json file containing questions and answers was provided to us. The model training was done on Google Collaboratory.In this report, we have mentioned the baseline models and the different changes made to that to reach our final model.

Baseline model

Image features and question features are extracted separately using CNN and RNN model respectively. After that, they are concatenated together and a dense layer that gives outputs using softmax activation function is used. We divided the data into train and test data in 0.7:0.3 proportion.

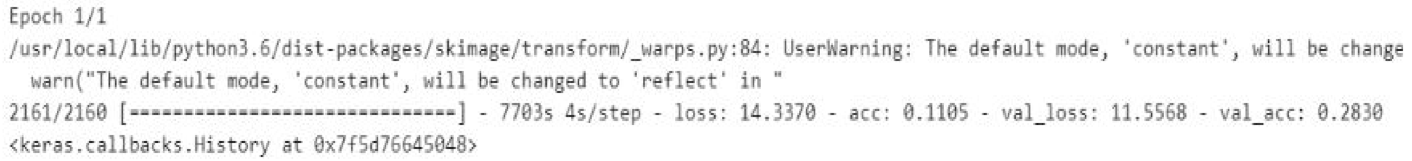
1. The json file is loaded to get the questions and answers for all images.
2. Each question is tokenized to a vector of length 200 units and the embedded in a dimension of (50,) using word embeddings from Glove file.
3. The pretrained standard VGG 16 model and the pretrained VGG16 CNN weights file is used.
4. The CNN weights file gives a 4096 dimension vector. All the images are resized to 224 x 224
5. Both the models are concatenated together to give a one hot encoded output.
7. ReLu activation function is used initially.
8. RMSProp optimizer is used.

Total Parameters: 134,260,544
Trainable parameters : 134,260,544
Non Trainable Parameters : 0

Concatenated Model

Total_params::135,473,834
Trainable_params::135,469,784
Non-trainable params: 4,050

Accuracy - 28%
Model Size - 950 MB



```
Epoch 1/1
/usr/local/lib/python3.6/dist-packages/skimage/transform/_warps.py:84: UserWarning: The default mode, 'constant', will be change
  warn("The default mode, 'constant', will be changed to 'reflect' in "
2161/2160 [==============================] - 7703s 4s/step - loss: 14.3370 - acc: 0.1105 - val_loss: 11.5568 - val_acc: 0.2830
<keras.callbacks.History at 0x7f5d76645048>
```

From this point onwards two broad objectives were pursued-

1)Increasing Accuracy

1) Glove 300dB text file is used instead of Glove 50dB text file for vector embedding in RNN model.
2) Adam Optimizer Used
3) Addition of MLP
4) Attention Mechanism added

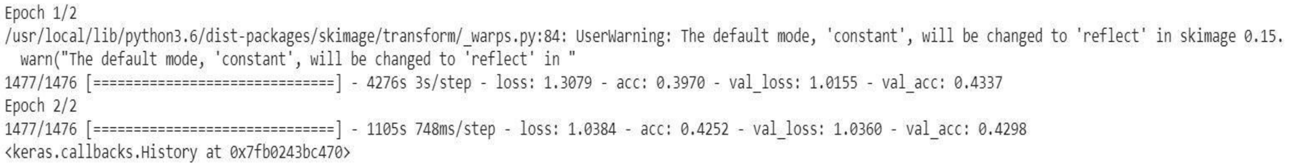
2)Reduction in model size

1. Pretrained VGG16 model is replaced by a CNN model written using ‘Relu’ activation function.
2. Images are resized to 32 x 32 from 224 x 224 to further reduce the size.
3. Removal of few dense layers from CNN model and from final layer

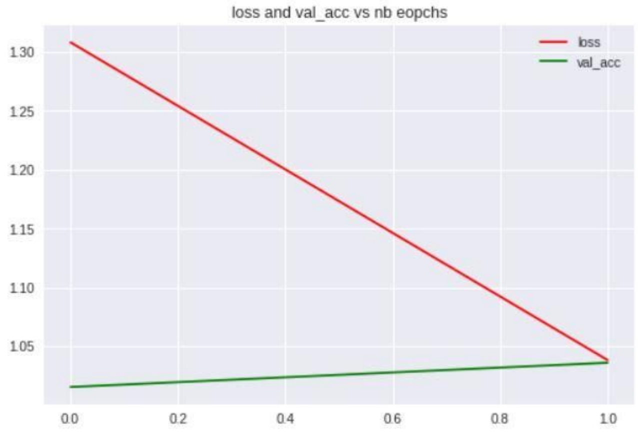
Addition of MLP

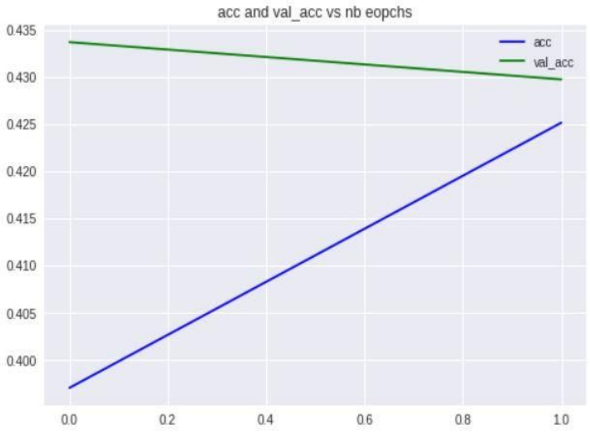
To further increase the accuracy of the concatenated model, two dense layers are added at the output of the merged model with 16 hidden units each. The **tanh** activation function is used with dropout rate of 0.5 . Here, the optimizer we used was rmsprop which gave the accuracy of 43% and model size was 1.5 MB. This blog post is used for reference. The final model gives a twenty-four featured output which selects the most probable output as the answer to the question.

Next we replaced LSTM in the baseline model with CuDNN LSTM and got an Accuracy of 42.98% model size of 5.89mb

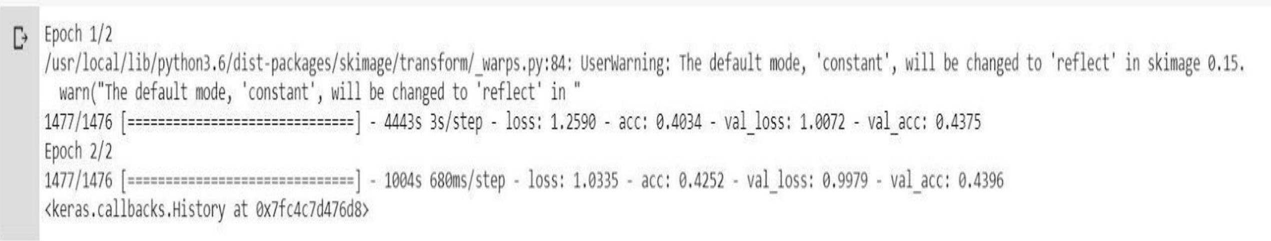


```
Epoch 1/2
/usr/local/lib/python3.6/dist-packages/skimage/transform/_warps.py:84: UserWarning: The default mode, 'constant', will be changed to 'reflect' in skimage 0.15.
  warn("The default mode, 'constant', will be changed to 'reflect' in "
1477/1476 [==============================] - 4276s 3s/step - loss: 1.3079 - acc: 0.3970 - val_loss: 1.0155 - val_acc: 0.4337
Epoch 2/2
1477/1476 [==============================] - 1105s 748ms/step - loss: 1.0384 - acc: 0.4252 - val_loss: 1.0360 - val_acc: 0.4298
<keras.callbacks.History at 0x7fb0243bc470>
```

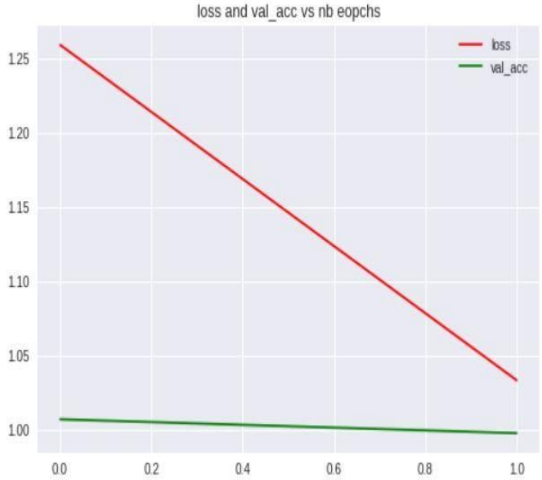


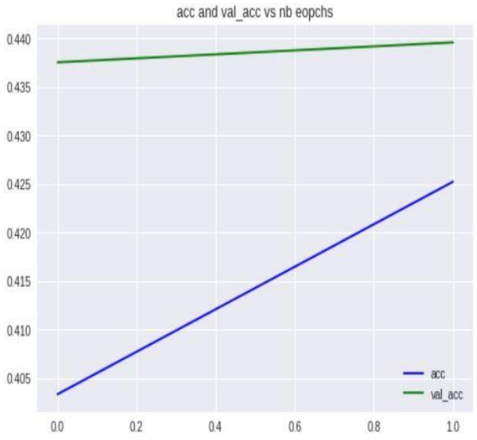


Next we used a custom CNN of 4 Convolutional layers instead of VGG net and also we removed stopwords from the questions. Reduced the max length from 200 to 100,added one Convolution layer to CNN and changed the activation function of 1st layer to tanh and also removed the second last layer from the RNN model. We got an accuracy of 43.96% with this model on the test dataset. Model size=5.89mb

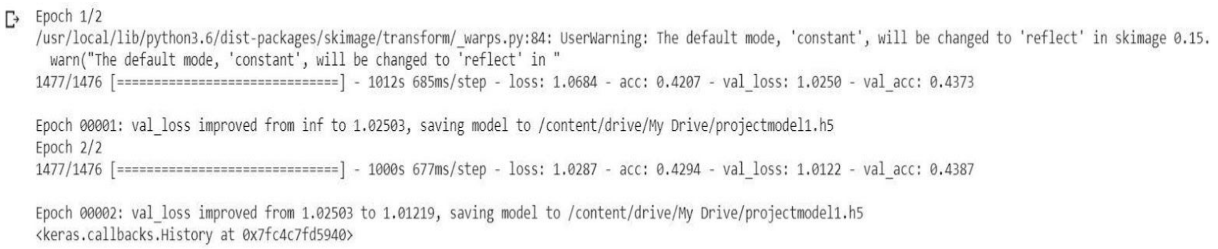


```
Epoch 1/2
/usr/local/lib/python3.6/dist-packages/skimage/transform/_warps.py:84: UserWarning: The default mode, 'constant', will be changed to 'reflect' in skimage 0.15.
  warn("The default mode, 'constant', will be changed to 'reflect' in "
1477/1476 [==============================] - 4443s 3s/step - loss: 1.2590 - acc: 0.4034 - val_loss: 1.0072 - val_acc: 0.4375
Epoch 2/2
1477/1476 [==============================] - 1004s 680ms/step - loss: 1.0335 - acc: 0.4252 - val_loss: 0.9979 - val_acc: 0.4396
<keras.callbacks.History at 0x7fc4c7d476d8>
```

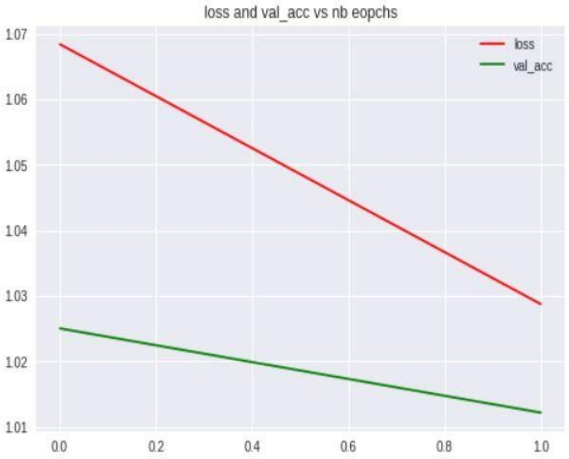


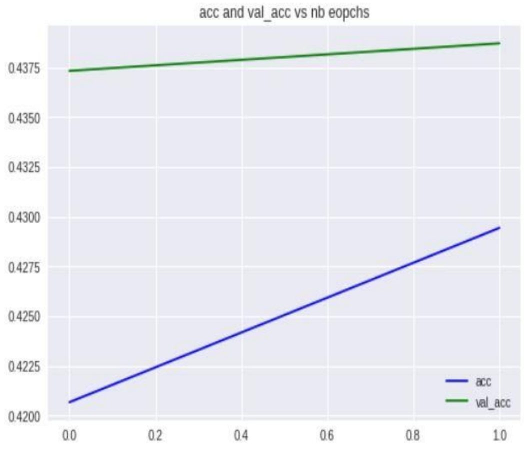


Next we in the second model instead of removing stop words we kept the questions as it is and got an Accuracy of 43.87% on test data and model size=3.51mb

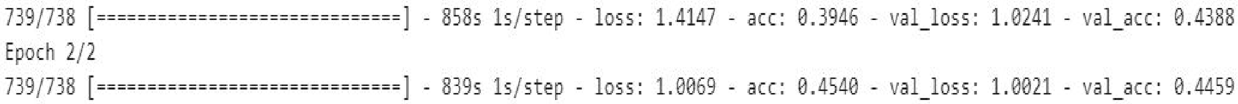


```
Epoch 1/2
/usr/local/lib/python3.6/dist-packages/skimage/transform/_warps.py:84: UserWarning: The default mode, 'constant', will be changed to 'reflect' in skimage 0.15.
  warn("The default mode, 'constant', will be changed to 'reflect' in "
1477/1476 [==============================] - 1012s 685ms/step - loss: 1.0684 - acc: 0.4207 - val_loss: 1.0250 - val_acc: 0.4373

Epoch 00001: val_loss improved from inf to 1.02503, saving model to /content/drive/My Drive/projectmodel1.h5
Epoch 2/2
1477/1476 [==============================] - 1000s 677ms/step - loss: 1.0287 - acc: 0.4294 - val_loss: 1.0122 - val_acc: 0.4387

Epoch 00002: val_loss improved from 1.02503 to 1.01219, saving model to /content/drive/My Drive/projectmodel1.h5
<keras.callbacks.History at 0x7fc4c7fd5940>
```

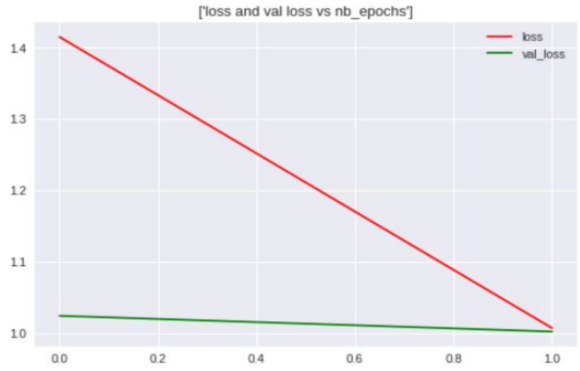


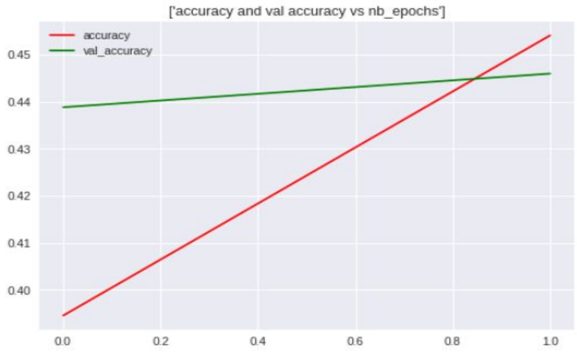


Next experiment was done with a CNN model having 3 convolutional layers (kernel size=16,32,64 followed by Pooling and one dense layer)and RNN model with one embedding layer and one lstm layer was used. This was merged and fed to another dense layer giving one hot encoded outputs
Model size=1.6 mB and accuracy=44.59%

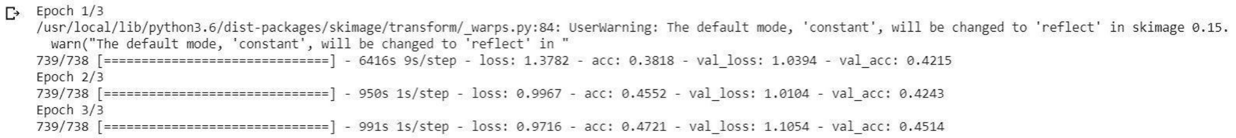


```
739/738 [==============================] - 858s 1s/step - loss: 1.4147 - acc: 0.3946 - val_loss: 1.0241 - val_acc: 0.4388
Epoch 2/2
739/738 [==============================] - 839s 1s/step - loss: 1.0069 - acc: 0.4540 - val_loss: 1.0021 - val_acc: 0.4459
```

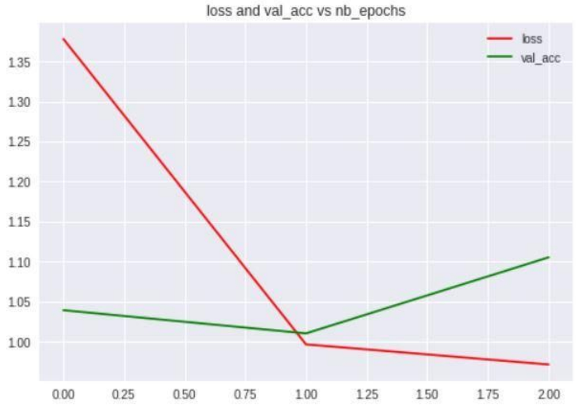




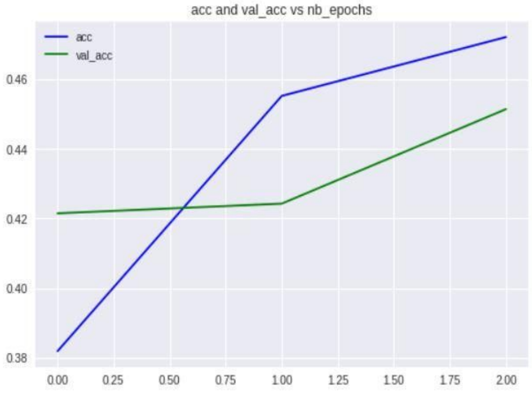
Next we added attention mechanism to the model and ran 3 epochs on it and got an accuracy of 45.14% with model size of 2.7mb



```
Epoch 1/3
/usr/local/lib/python3.6/dist-packages/skimage/transform/_warps.py:84: UserWarning: The default mode, 'constant', will be changed to 'reflect' in skimage 0.15.
  warn("The default mode, 'constant', will be changed to 'reflect' in "
739/738 [==============================] - 6416s 9s/step - loss: 1.3782 - acc: 0.3818 - val_loss: 1.0394 - val_acc: 0.4215
Epoch 2/3
739/738 [==============================] - 950s 1s/step - loss: 0.9967 - acc: 0.4552 - val_loss: 1.0104 - val_acc: 0.4243
Epoch 3/3
739/738 [==============================] - 991s 1s/step - loss: 0.9716 - acc: 0.4721 - val_loss: 1.1054 - val_acc: 0.4514
```



loss and val_acc vs nb_epochs



acc and val_acc vs nb_epochs