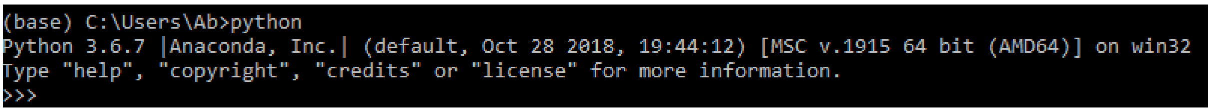
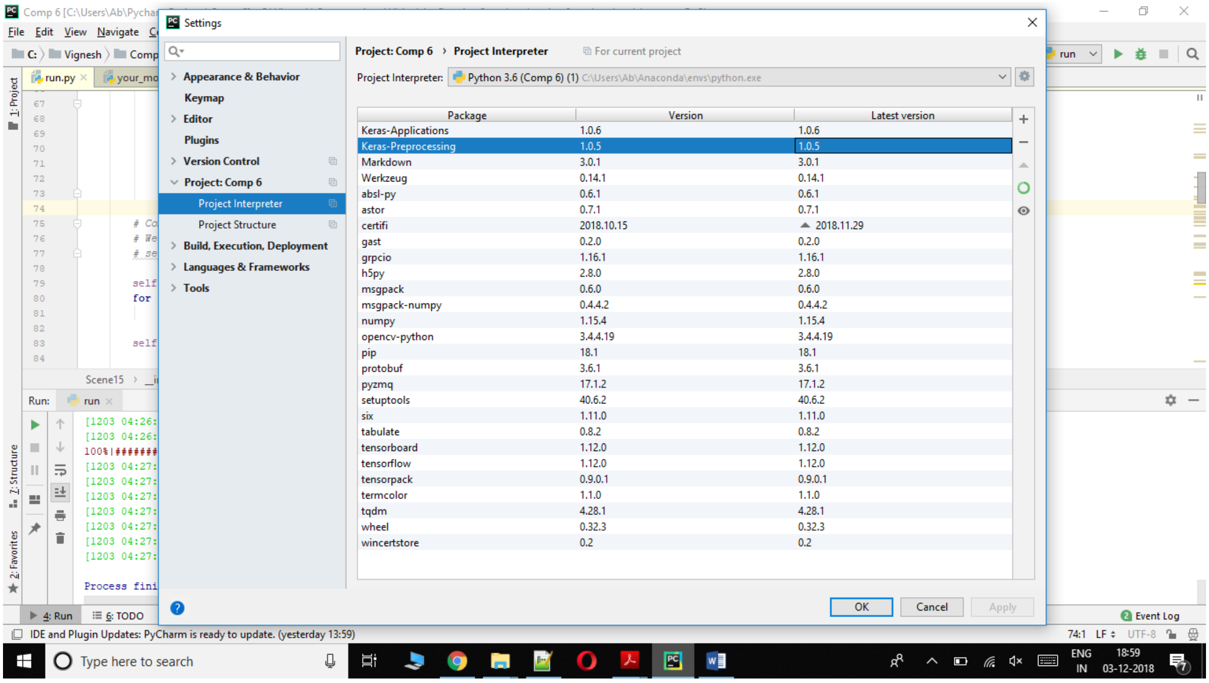
3.1 TensorFlow installation.

- I have installed the python 3.6.7 version in windows OS and specific packages such as tensor flow, tensorpack, opencv-python, tensorboard have been installed. I have used the Pycharm IDE for running the model and testing.
- The conda environment has been setup in Pycharm using the Project Interpreter and have shown all the installed packages which was done for this project.
- I have used both the CPU and GPU environments separately for two different models for testing and training them.

Figure 1 and 2 shows the installation of python and other packages which was installed for this project.



```
(base) C:\Users\Ab>python
Python 3.6.7 |Anaconda, Inc.| (default, Oct 28 2018, 19:44:12) [MSC v.1915 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```



Project: Comp 6 › Project Interpreter For current project

Project Interpreter: Python 3.6 (Comp 6) (1) C:\Users\Ab\Anaconda\envs\python.exe

Package	Version	Latest version
Keras-Applications	1.0.6	1.0.6
Keras-Preprocessing	1.0.5	1.0.5
Markdown	3.0.1	3.0.1
Werkzeug	0.14.1	0.14.1
absl-py	0.6.1	0.6.1
astor	0.7.1	0.7.1
certifi	2018.10.15	▲ 2018.11.29
gast	0.2.0	0.2.0
grpcio	1.16.1	1.16.1
h5py	2.8.0	2.8.0
msgpack	0.6.0	0.6.0
msgpack-numpy	0.4.4.2	0.4.4.2
numpy	1.15.4	1.15.4
opencv-python	3.4.4.19	3.4.4.19
pip	18.1	18.1
protobuf	3.6.1	3.6.1
pyzmq	17.1.2	17.1.2
setuptools	40.6.2	40.6.2
six	1.11.0	1.11.0
tabulate	0.8.2	0.8.2
tensorboard	1.12.0	1.12.0
tensorflow	1.12.0	1.12.0
tensorpack	0.9.0.1	0.9.0.1
termcolor	1.1.0	1.1.0
tqdm	4.28.1	4.28.1
wheel	0.32.3	0.32.3
wincertstore	0.2	0.2

IDE and Plugin Updates: PyCharm is ready to update. (yesterday 13:59)

3.2 Training from scratch

3.2.1 Improvements in the model

- Feature normalisation (Image standardization) has been performed by subtracting the image input with the mean of the input image and then dividing the standard deviation of the image input.
- Dropout regularization has been added in my model by calling the tf.nn.dropout() function by passing the logits calculated from the Fully connected layer.
- Data augmentation has been performed by random cropping the image using the imgaug.Resize() function. Once the cropping is done, the image has been resized to the original image.
- The architecture has been changed by adding two convolution layers to the image, keeping the kernel size to 3*3 and saved.

- At every convolution layer, max pooling has been performed in the 2*2 window with its input being the output of the previous convolution layer.
- Fully Connected(FC) layer has been introduced here with the output received from the maxpooling as the FC layer input.
- The dropout has been added by getting the output of FC layers as it's input.
- The FC layer is called again after the dropout is performed and the output is sent to calculate the cost error.

Training your_model

- After making the above changes, I have trained and tested my model by running in Pycharm.
- In the edit configurations sections I have provided the path of the run.py file in the Script path and provided the --task 1--gpu -1 (for CPU) for the execution.
- I ran the model in CPU and have provided you the test results below. **Achieved 50 percent test accuracy and 99 percent training accuracy by fine tuning the hyperparameter.**
- Log is saved and provided in the folder logs/task 1/test_log for the model ran in CPU.

3.2.2Tensorboard visualize steps

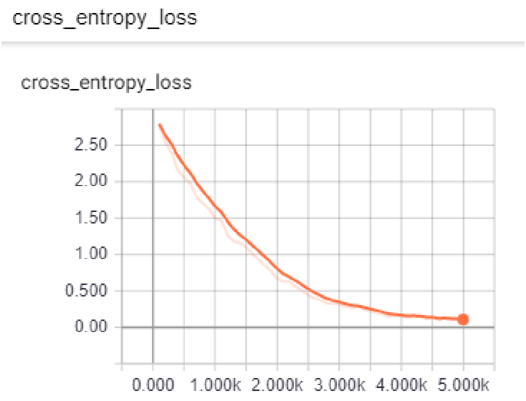
- The tensorboard package has been installed and ran the below query to visualize the logs that are saved for the epochs ran for the model.
  tensorboard --logdir=train_log/run

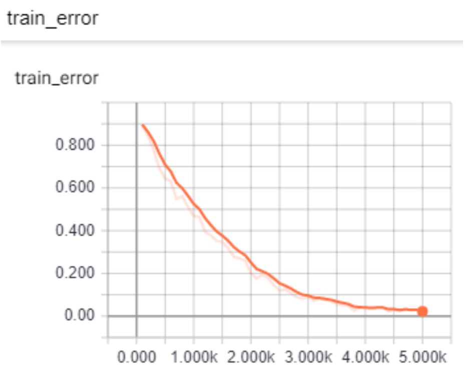
- Navigated to this URL http://localhost:6006/ and received the below training and validation error.

Tensorboard graph for Your_model

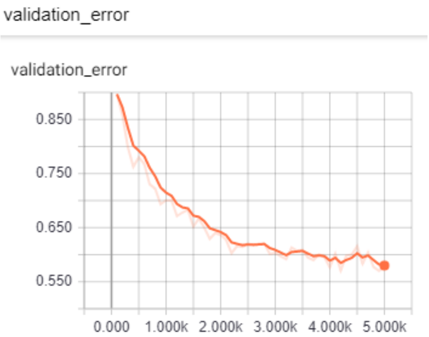
1) Tensorboard graph- In the hyperparameters file I have given the Epochs to 50, Batch size-to 50, learning rate to 0.01. Received testing accuracy of 46 percent.

Cross Entropy Loss Training Loss



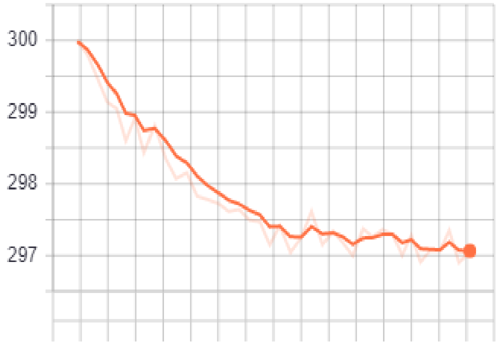


Validation Loss

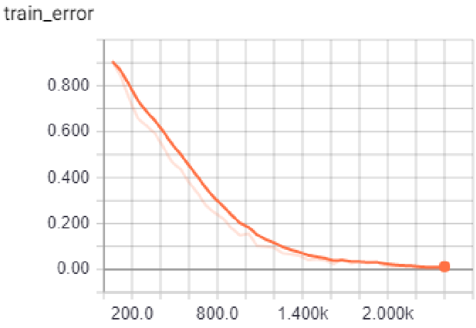


2) Improved the accuracy by fine tuning the hyperparameters. Changed the Epochs to 40, Batch size to 25, learning rate as 0.01 and **achieved testing accuracy of 50 percent. Log folder is available in the path /logs/task 1/test_log**

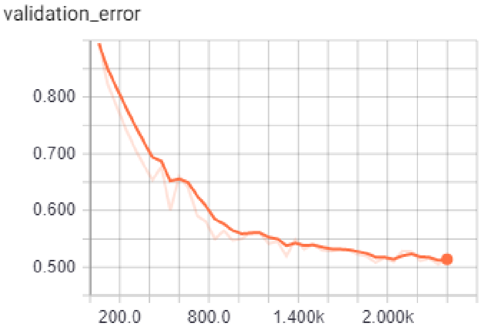
Cross Entropy Loss



Training Error



Validation Error



3.3 Fine Tuning the VGG

3.3.1) Fine tuning the parameters for VGG model.

- The VGG model has been downloaded and placed under the /code path.
- Hyperparameters are tweaked such that in hyperparameters file I have provided the batch size to 5 and the learning rate as 0.00001.

Training vgg_model

- After making the above changes, I have trained and tested my model by running in Pycharm.
- In the edit configurations sections I have provided the path of the run.py file in the Script path and provided the --task 2 –gpu 0 (for GPU) for the execution.
- I ran the model in GPU and have provided you the test results below.
- Achieved 86 percent test accuracy and 97 percent training accuracy by fine tuning the hyperparameter.
- Image standardisation was performed while training the vgg model that has improved my testing accuracy to 86.6 percent.

3.3.2) Tensorboard visualize steps

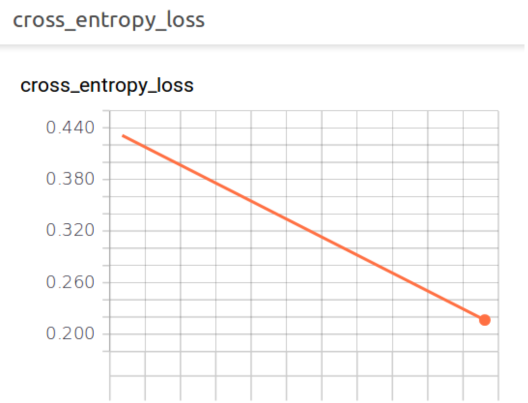
- The tensorboard package has been installed and ran the below query to visualize the logs that are saved for the epochs ran for the model.
  tensorboard --logdir=train_log/run

- Navigated to this URL **http://localhost:6006/** and received the below training and validation error.

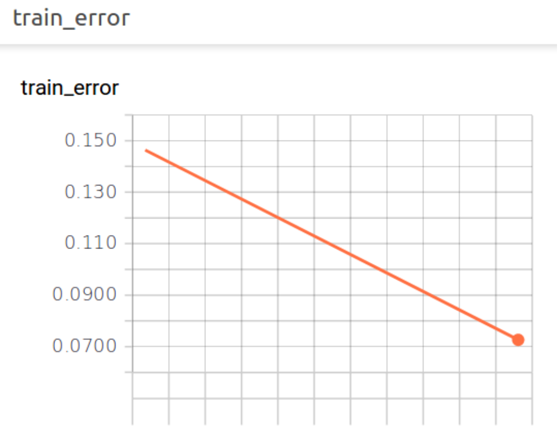
Tensorboard graph for vgg_model

1) Tensorboard graph- Epochs 5, Batch size- 5, learning rate 0.00001. Received an accuracy of 86 percent. **I have provided the log file I ran using GPU for 5 epochs and received an accuracy of 86%.** The training accuracy is achieved up to 99.4 percent and the validation accuracy is achieved up to 86.8 percent accuracy. I have provided the log file in the path /logs/task 2/test_log/log
2) After performing the image standardisation, 2 epochs have been run and **achieved the testing accuracy of 86.8%** with the training accuracy of 99.3% provided the results as below. The log file for this has been placed under /logs/task 2/test_log/log_1

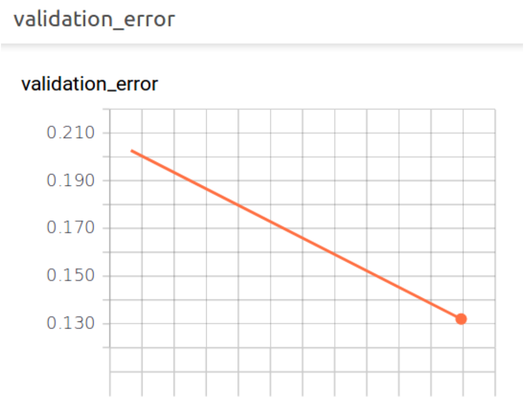
Cross Entropy Loss



Training Loss



Validation Loss


validation_error
validation_error