FAQ - Transfer Learning

#### **1. Are there any other Transfer Learning or pre-trained models other than VGG16?**

Yes, there are other pre-trained models like [ResNet](https://www.analyticsvidhya.com/blog/2021/08/how-to-code-your-resnet-from-scratch-in-tensorflow/), [EfficientNet](https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/), [InceptionNet](https://www.analyticsvidhya.com/blog/2018/10/understanding-inception-network-from-scratch/), and [MobileNet](https://www.geeksforgeeks.org/image-recognition-with-mobilenet/) architectures that can be implemented in Tensorflow.

#### **2. Why should we use Transfer Learning and fine-tune our model?**

* The reason is, through experimentation, we observed that early layers of CNN collect basic elements of images such as borders, lines, and so on, whereas subsequent layers capture more specific features such as faces and object shapes. If we use transfer learning, we will need to train the last few layers or the output classifying layer only.
* A pre-trained model is trained with datasets that are most likely bigger than your dataset. Also, because of reason number 1, and we have trained it with a bigger dataset, we have the first few layers that are generalized and hence this helps to reduce overfitting.
* Since we are only backpropagating within the last few layers, it saves our time and effort.
* Architecture-wise, people spend time researching how every feature interacts with an image and hence it is recommended to use this architecture, and published/publicly known result implies that it gives good result.

#### **3. Why should we use GPUs for training the deep learning and the Transfer Learning models?**

GPUs can perform multiple, simultaneous computations. This enables the distribution of training processes and can significantly speed up machine learning operations. With GPUs, we can accumulate many cores that use fewer resources without sacrificing efficiency or power.

#### **4. How can we add labels to both the x-axis and y-axis while plotting the confusion matrix?**

We should use the set\_ticklabels() function to set the labels for both axes. We can use the below code to set the labels.

*# Obtaining the categorical values from y\_test\_encoded and y\_pred*

*y\_pred\_arg=np.argmax(y\_pred,axis=1)*

*y\_test\_arg=np.argmax(y\_test\_encoded,axis=1)*

*# Plotting the Confusion Matrix using confusion matrix() function which is also predefined tensorflow module*

*confusion\_matrix = tf.math.confusion\_matrix(y\_test\_arg,y\_pred\_arg)*

*f, ax = plt.subplots(figsize=(10, 8))*

*sns.heatmap(*

*confusion\_matrix,*

*annot=True,*

*linewidths=.4,*

*fmt="d",*

*square=True,*

*ax=ax*

*)*

*# labels, title and ticks*

*ax.set\_xlabel('Predicted labels');ax.set\_ylabel('True labels');*

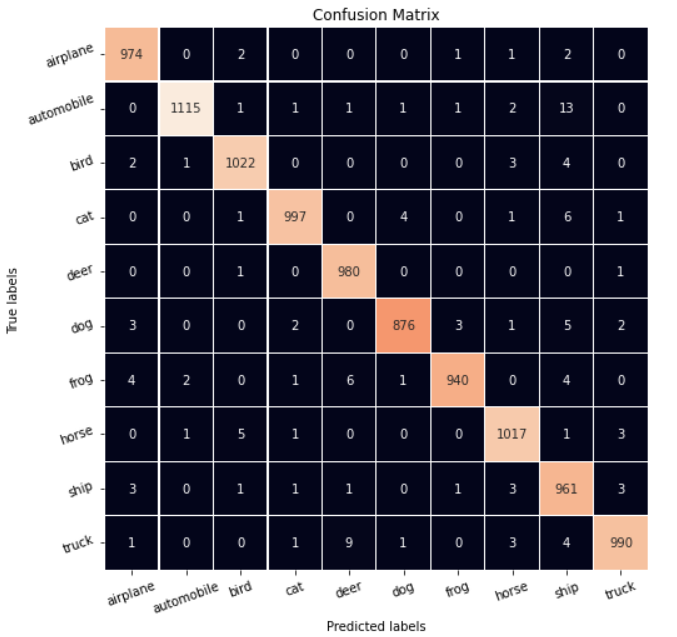
*ax.set\_title('Confusion Matrix');*

*ax.xaxis.set\_ticklabels(["airplane", "automobile", "bird", "cat", "deer", "dog", "frog", "horse", "ship", "truck"],rotation=20)*

*ax.yaxis.set\_ticklabels(["airplane", "automobile", "bird", "cat", "deer", "dog", "frog", "horse", "ship", "truck"],rotation=20)*

*plt.show()*

*The output can be visualized as shown below:*

**