In[]:
```
Name:Devika Sawant RollNo:54 BE IT AssignmentNo:6
```

In[]:
```
Title-ObjectdetectionusingTransferLearningofCNNarchitectures
a. Loadinapre-trainedCNNmodeltrainedonalargedataset
b. Freezeparameters(weights)inmodel’slowerconvolutionallayers
c. Addcustomclassifierwithseverallayersoftrainableparameterstomodel
d. Trainclassifierlayersontrainingdataavailablefortask
e. Fine-tunehyperparametersandunfreezemorelayersasneeded
```

In[1]:
```
importtensorflowastf
fromtensorflowimportkeras
fromtensorflow.keras.applicationsimportVGG16
fromtensorflow.keras.modelsimportModel
fromtensorflow.keras.layersimportDense,Flatten
fromtensorflow.keras.optimizersimportAdam
fromtensorflow.keras.preprocessing.imageimportImageDataGenerator
importnumpyasnp
```

```
C:\Users\sanka\anaconda3\lib\site-packages\requests\init.py:89:       RequestsDepen
dencyWarning: urllib3 (2.2.3) or chardet (3.0.4) doesn't match a supported versio
n!
  warnings.warn("urllib3({})orchardet({})doesn'tmatchasupported"
```

In[2]:
```
train_dir="C:/Users/sanka/OneDrive/Desktop/Datasets/cifar-10-img/cifar-10-img
test_dir="C:/Users/sanka/OneDrive/Desktop/Datasets/cifar-10-img/cifar-10-img/
```

In[3]:
```
train_dir
```

Out[3]:
```
'C:/Users/sanka/OneDrive/Desktop/Data    sets/cifar-10-img/cifar-10-img/train'
```

In[4]:
```
test_dir
```

Out[4]:
```
'C:/Users/sanka/OneDrive/Desktop/Datasets/cifar-10-img/cifar-10-img/test'
```

In[5]:
```
train_datagen = ImageDataGenerator(
    rescale=1.0 / 255,
)

test_datagen = ImageDataGenerator(
    rescale=1.0 / 255,
)
```

In[6]:
```
#herebatch_sizeisthenumberofimagesineachbatch

train_batch_size=5000
train_generator=train_datagen.flow_from_directory(
    train_dir,
    target_size=(32, 32),
    batch_size=train_batch_size,
    class_mode='categorical'
)

test_batch_size=1000
test_generator=test_datagen.flow_from_directory(
    test_dir,
    target_size=(32, 32),
    batch_size=test_batch_size,
    class_mode='categorical'
)
```

```
Found40079imagesbelongingto10classes.
Found9921imagesbelongingto10classes.
```

In[7]:

```
x_train,y_train=train_generator[0] x_test,
y_test = test_generator[0]

print(len(x_train))
print(len(x_test))
```

```
5000
1000
```

In[8]:

```
#LoadVGG16withouttoplayers
weights_path="vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5"
base_model=VGG16(weights=weights_path,include_top=False,input_shape=(32,32,
```

In[9]:

```
for layer in base_model.layers:
   layer.trainable = False
```

In[10]:

```
x=Flatten()(base_model.output)
x = Dense(256, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3)(x)x
= Dense(256, activation='relu')(x) x
= tf.keras.layers.Dropout(0.3)(x)
predictions=Dense(10,activation='softmax')(x)

#Createthemodel
model=Model(inputs=base_model.input,outputs=predictions)
#Compilethemodel
model.compile(optimizer="adam",loss='categorical_crossentropy',metrics=['accur
```

In[11]:

```
#Trainthemodel
model.fit(x_train,y_train,batch_size=64,epochs=10,validation_data=(x_test,y
Epoch1/10
79/79[==============================]-20s     249ms/step  -loss:  1.9585  -accurac
y:0.3034-val_loss:1.5953-val_accuracy:      0.4590
Epoch2/10
79/79[==============================]-18s     223ms/step  -loss:  1.5787  -accurac
y:0.4440-val_loss:1.4498-val_accuracy:      0.4990
Epoch3/10
79/79[==============================]-18s     224ms/step  -loss:  1.4612  -accurac
y:0.4852-val_loss:1.3657-val_accuracy:      0.5410
Epoch 4/10
79/79[==============================]-19s     235ms/step  -loss:  1.3683  -accurac
y:0.5156-val_loss:1.3115-val_accuracy:      0.5580
Epoch5/10
79/79[==============================]-16s     197ms/step  -loss:  1.2856  -accurac
y:0.5424-val_loss:1.3732-val_accuracy:      0.5240
Epoch6/10
79/79[==============================]-16s     199ms/step  -loss:  1.2533  -accurac
y:0.5616-val_loss:1.3063-val_accuracy:      0.5550
Epoch7/10
79/79[==============================]-16s     198ms/step  -loss:  1.1820  -accurac
y:0.5876-val_loss:1.2719-val_accuracy:      0.5610
Epoch 8/10
79/79[==============================]-17s     209ms/step  -loss:  1.1252  -accurac
y:0.6130-val_loss:1.2695-val_accuracy:      0.5690
Epoch 9/10
79/79[==============================]-18s     229ms/step  -loss:  1.0940  -accurac
y:0.6174-val_loss:1.2686-val_accuracy:      0.5660
Epoch10/10
79/79[==============================]-18s     232ms/step  -loss:  1.0463  -accurac
y:0.6370-val_loss:1.2713-val_accuracy:0.5650
```

Out[11]:
```
<keras.callbacks.Historyat0x2997785e520>
```

In[12]:
```
base_model=VGG16(weights=weights_path,include_top=False,input_shape=(32,32,
#freezealllayersfirst
for layer in base_model.layers:
   layer.trainable = False
#unfreezelast4layersofbasemodel
for layer in base_model.layers[len(base_model.layers) - 4:]:
   layer.trainable = True
#fine-tuninghyperparameters
x=Flatten()(base_model.output)
x = Dense(256, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3)(x)x
= Dense(512, activation='relu')(x) x
= tf.keras.layers.Dropout(0.3)(x)
predictions=Dense(10,activation='softmax')(x)

#Createthemodel
model=Model(inputs=base_model.input,outputs=predictions)
#Compilethemodel
model.compile(optimizer=Adam(learning_rate=0.001),loss='categorical_crossentrop
#trainingfinetunedmodel
model.fit(x_train,y_train,batch_size=64,epochs=10,validation_data=(x_test,y
Epoch1/10
79/79[==============================]-49s     608ms/step  -loss:  1.9784  -accurac
y:0.2556-val_loss:1.4228-val_accuracy:      0.5060
Epoch2/10
79/79[==============================]-48s     612ms/step  -loss:  1.3392  -accurac
y:0.5314-val_loss:1.2556-val_accuracy:      0.5620
Epoch3/10
79/79[==============================]-46s     589ms/step  -loss:  1.0812  -accurac
y:0.6280-val_loss:1.2446-val_accuracy:      0.6020
Epoch 4/10
79/79[==============================]-46s     588ms/step  -loss:  0.9124  -accurac
y:0.6864-val_loss:1.1299-val_accuracy:      0.6360
Epoch5/10
79/79[==============================]-46s     589ms/step  -loss:  0.7689  -accurac
y:0.7438-val_loss:1.1596-val_accuracy:      0.6400
Epoch6/10
79/79[==============================]-47s     601ms/step  -loss:  0.6404  -accurac
y:0.7838-val_loss:1.1553-val_accuracy:      0.6500
Epoch7/10
79/79[==============================]-46s     581ms/step  -loss:  0.6092  -accurac
y:0.7936-val_loss:1.3129-val_accuracy:      0.6320
Epoch 8/10
79/79[==============================]-46s     584ms/step  -loss:  0.5241  -accurac
y:0.8254-val_loss:1.1735-val_accuracy:      0.6770
Epoch 9/10
79/79[==============================]-46s     581ms/step  -loss:  0.4470  -accurac
y:0.8554-val_loss:1.3096-val_accuracy:      0.6660
Epoch10/10
79/79[==============================]-46s     585ms/step  -loss:  0.3632  -accurac
y:0.8806-val_loss:1.5076-val_accuracy:0.6430
```

Out[12]:
```
<keras.callbacks.Historyat0x29901181ee0>
```

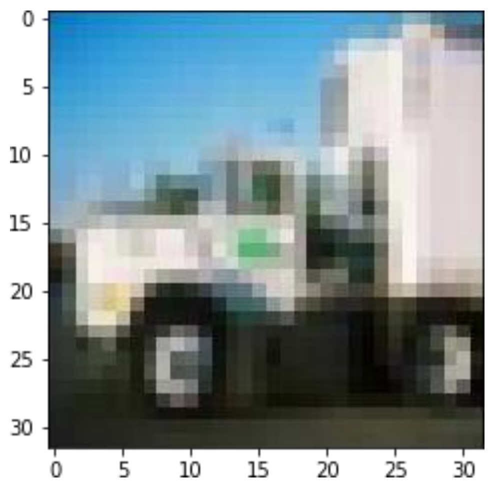
In[13]:
```
import matplotlib.pyplot as plt
predicted_value = model.predict(x_test)
```

```
32/32[==============================]-3s96ms/step
```

In[14]:
```
labels=list(test_generator.class_indices.keys())
```

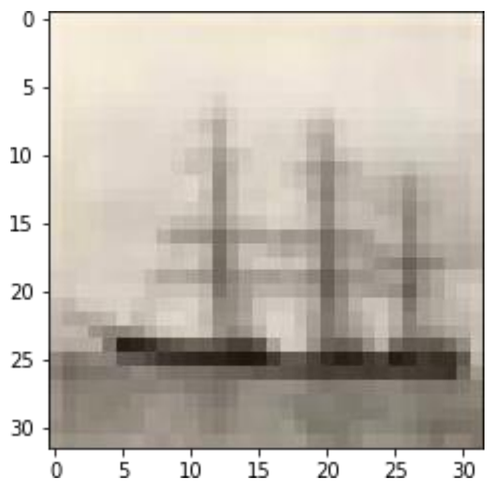
In[17]:
```
n=945
plt.imshow(x_test[n])
print("Preditcted:",labels[np.argmax(predicted_value[n])])
print("Actual:",labels[np.argmax(y_test[n])])
```

```
Preditcted:truck
Actual:truck
```



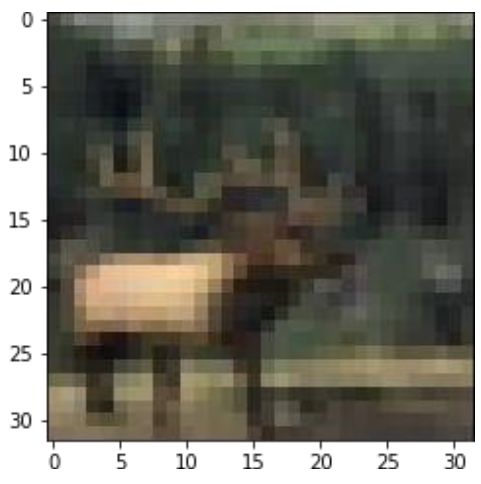
In[18]:
```
n=9
plt.imshow(x_test[n])
print("Preditcted:",labels[np.argmax(predicted_value[n])])
print("Actual:",labels[np.argmax(y_test[n])])
```

```
Preditcted:horse
Actual:ship
```



In[20]:
```
n=5
plt.imshow(x_test[n])
print("Preditcted:",labels[np.argmax(predicted_value[n])])
print("Actual:",labels[np.argmax(y_test[n])])
```

```
Preditcted:deer
Actual:deer
```



In[]:

```
```