Base model without modification:

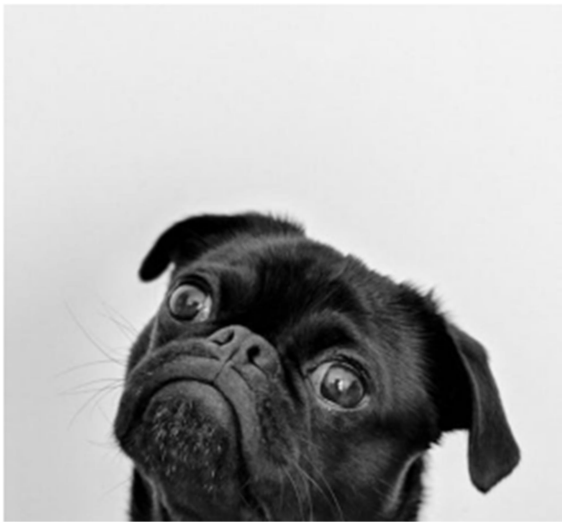
```
[11] print(test_acc)

     0.7085999846458435
```


```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
```

Predicted: truck

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
```

Predicted: cat

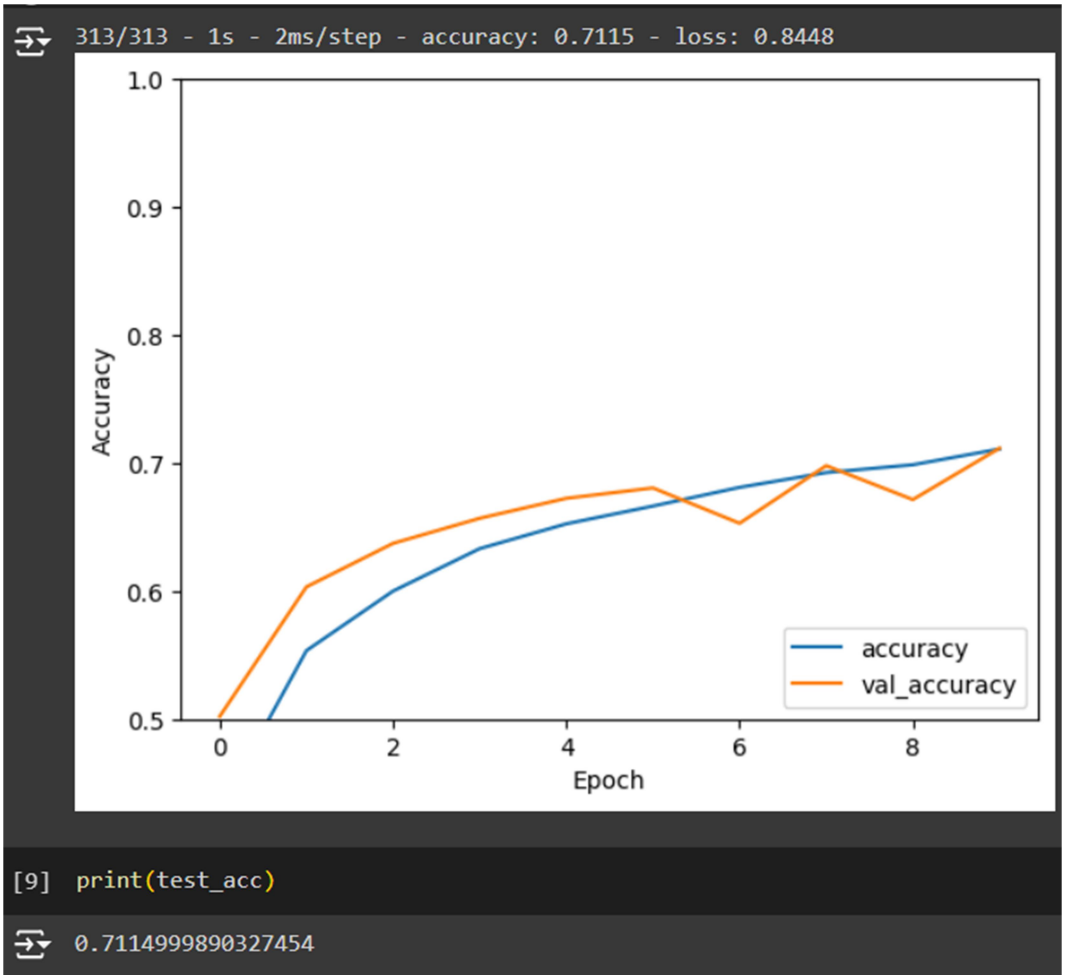


Predicted: frog

Code modification:

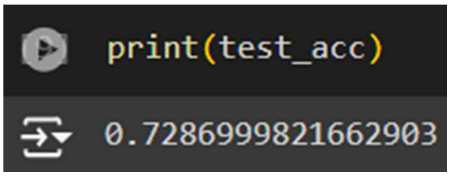
1) Add data augmentation including rotating image and horizontally flipping the image

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(
    rotation_range=20,
    horizontal_flip=True)

datagen.fit(train_images)
```



Result: the accuracy only go up by ~1%

2) Increase epoch number to 20



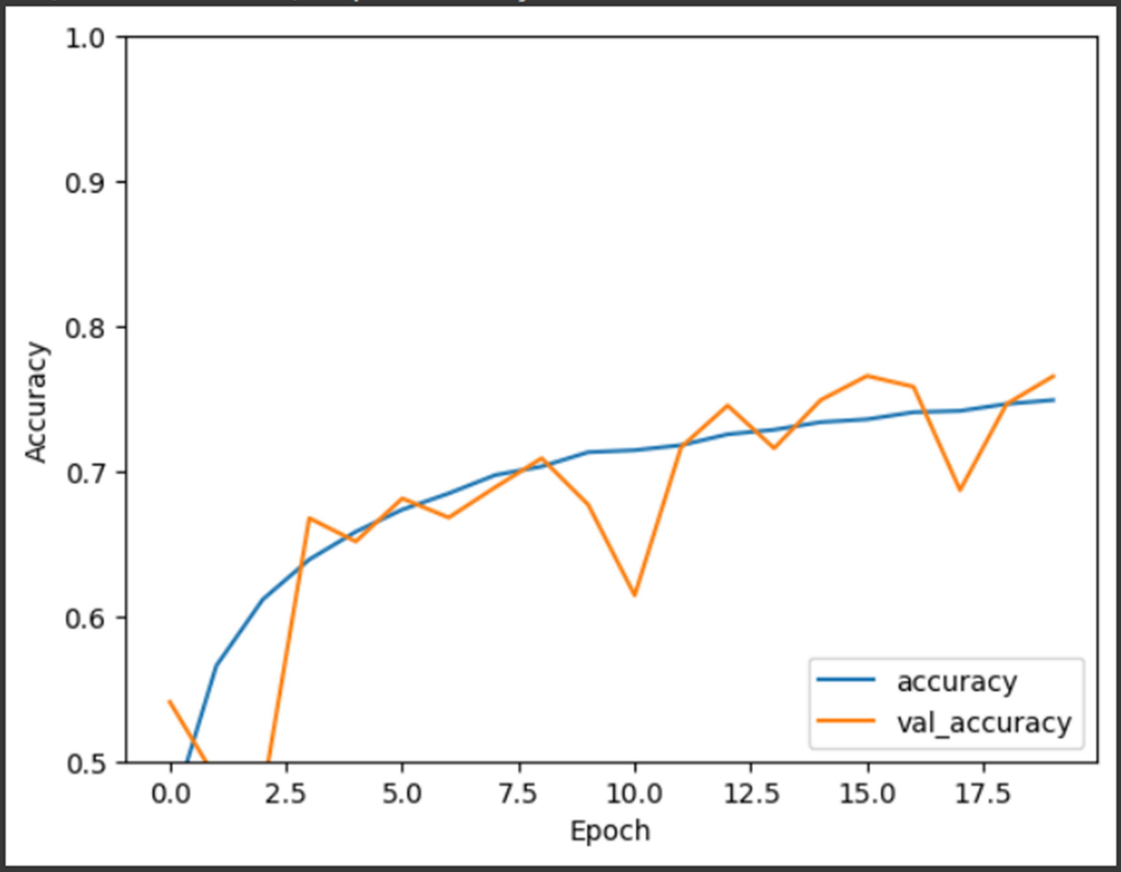
Result: this give me an additional 1%

3) Add Batch norm after a convo layer and add a dropout layer after 2 max pooling layer. I added more drop out layer but the result got worse and worse

```
model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(layers.BatchNormalization())
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.BatchNormalization())
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Dropout(0.3))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.BatchNormalization())
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())

model.add(layers.Dense(128, activation='relu'))
model.add(layers.BatchNormalization())

model.add(layers.Dense(10))

model.summary()
```

```
test_loss, test_acc = model.evaluate(test_images,  test_labels, verbose=2)
```

```
313/313 - 4s - 12ms/step - accuracy: 0.7654 - loss: 0.6730
```

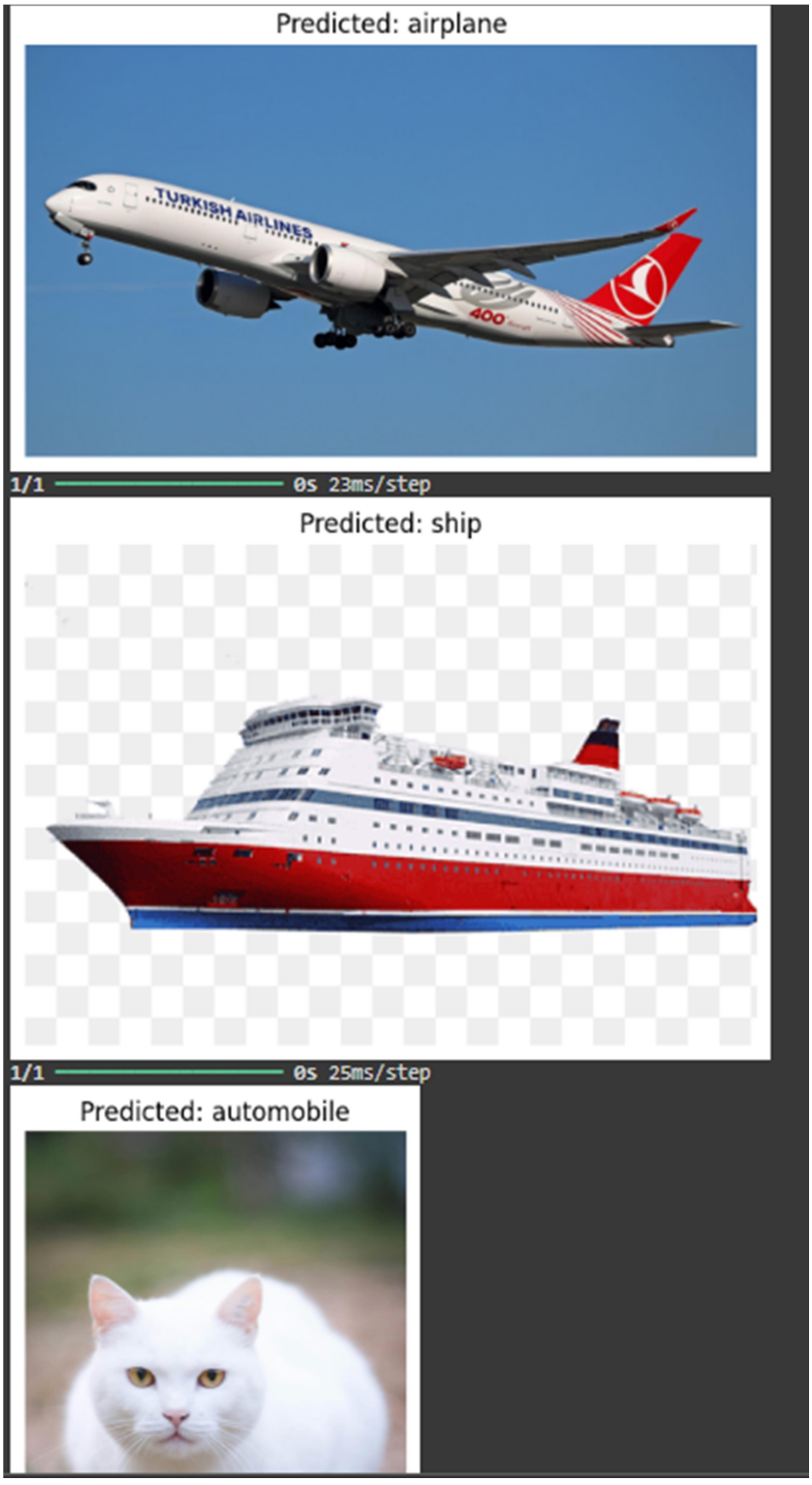


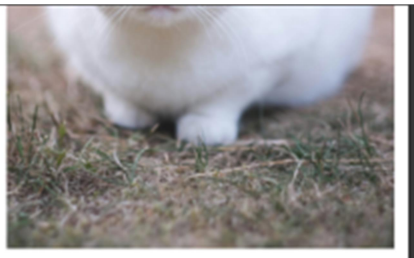

```
[ ] print(test_acc)
```

```
0.7653999924659729
```

4) Final validation

The result got worse ☹ compare to when I only made data augmentation to be honest





```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step
```

Predicted: automobile

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step
```

Predicted: frog