Reference solution for assignment 2

1. Consider the Cats & Dogs example. Start initially with a training sample of 1000, a validation sample of 500, and a test sample of 500 (like in the text). Use any technique to reduce overfitting and improve performance in developing a network that you train from scratch. What performance did you achieve?

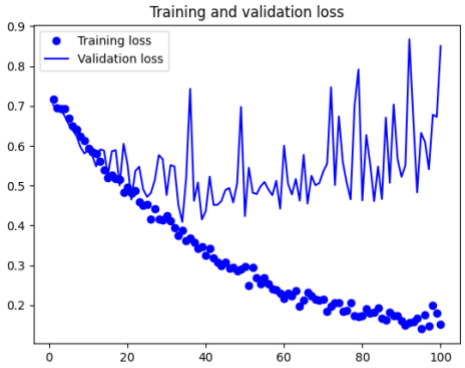
   Ans:

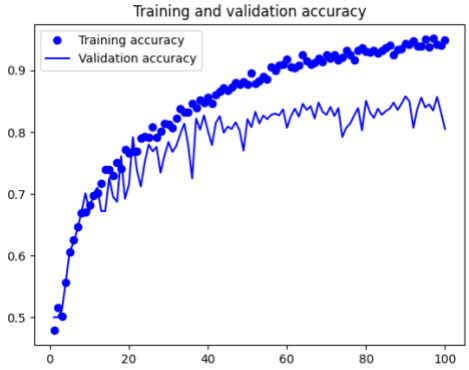
```
make_subset("train", start_index=0, end_index=1000) # 1000 training
set
make_subset("validation", start_index=1000, end_index=1500) # 500
validation set
make_subset("test", start_index=1500, end_index=2000) # 500 test set
```

   After adding augmentation and dropout layers,

```
data_augmentation = keras.Sequential(
    [
        layers.RandomFlip("horizontal"),
        layers.RandomRotation(0.1),
        layers.RandomZoom(0.2),
    ]
)
```

```
inputs = keras.Input(shape=(180, 180, 3))
x = data_augmentation(inputs)
x = layers.Rescaling(1./255)(x)
x = layers.Conv2D(filters=32, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=64, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=128, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=256, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=256, kernel_size=3, activation="relu")(x)
x = layers.Flatten()(x)
x = layers.Dropout(0.5)(x)
outputs = layers.Dense(1, activation="sigmoid")(x)
```





Test accuracy: 0.8040, Test loss: 0.4384
Other techniques like regularization layers can also be added to improve accuracy.

2. Increase your training sample size. You may pick any amount. Keep the validation and test samples the same as above. Optimize your network (again training from scratch). What performance did you achieve?

   Ans: **For example, we increase the size of training set to 3000.**

```
make_subset("train", start_index=0, end_index=3000) # increase the
size of training set to 3000
make_subset("validation", start_index=3000, end_index=3500) # 500
validation set
make_subset("test", start_index=3500, end_index=4000) # 500 test set
```

After adding augmentation and dropout layers,

Test accuracy: 0.8850, Test loss: 0.3337

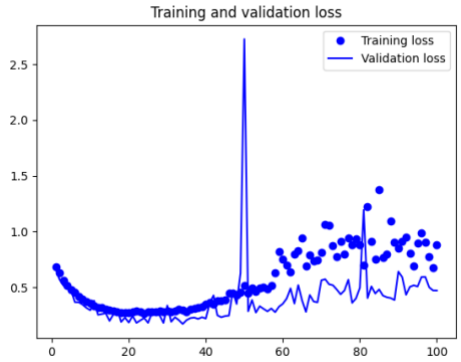
Conclusion: Increasing the training set size from 1000 to 3000 significantly improved the test accuracy from 0.804 to 0.885.

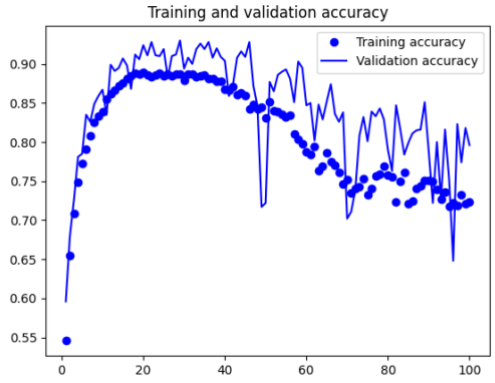
3. Now change your training sample so that you achieve better performance than those from Steps 1 and 2. This sample size may be larger, or smaller than those in the previous steps. The objective is to find the ideal training sample size to get best prediction results.

   Ans: For example, we **increase the size of training set to 6000**.

```
make_subset("train", start_index=0, end_index=6000) # increase
training size to 6000
make_subset("validation", start_index=6000, end_index=6500)
make_subset("test", start_index=6500, end_index=7000)
```

After adding augmentation and dropout layers,





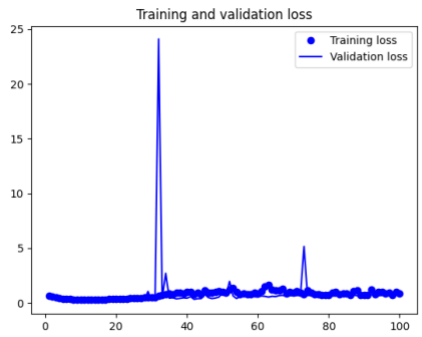
Test accuracy: 0.9230 Test loss: 0.1858

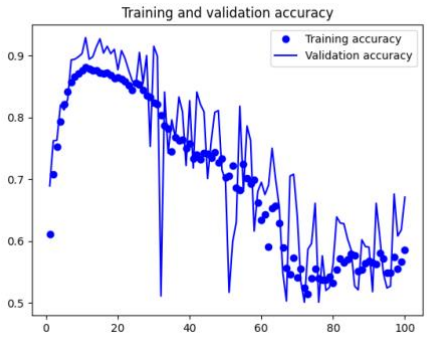
A more extreme example: use the complete dataset, i.e. increase the size of training set to 11500.

```
make_subset("train", start_index=0, end_index=11500)
make_subset("validation", start_index=11500, end_index=12000) # 500 validation set
make_subset("test", start_index=12000, end_index=12500) # 500 test set
```

```
Test accuracy: 0.887, Test loss: 0.265
```

After adding augmentation and dropout layers,





Test accuracy: 0.9320, Test loss: 0.1555

Conclusion: Increasing the training set size generally does not hurt an algorithm's performance, and it may help significantly.

For example, increasing the training set size from 3,000 to 6,000 significantly improved the test accuracy from 0.885 to 0.923. However, increasing the training set size from 6,000 to 115,000 only slightly improved the test accuracy by around 1%.

4. Repeat Steps 1-3, but now using a pretrained network. The sample sizes you use in Steps 2 and 3 for the pretrained network may be the same or different from those using the network where you trained from scratch. Again, use any and all optimization techniques to get best performance.

   Step1: training set with size 1000

   a. Feature extraction with a pretrained model with data augmentation and dropout layers

```
conv_base  = keras.applications.vgg16.VGG16(
    weights="imagenet",
    include_top=False)
conv_base.trainable = False
```

```
data_augmentation = keras.Sequential(
    [
        layers.RandomFlip("horizontal"),
        layers.RandomRotation(0.1),
        layers.RandomZoom(0.2),
    ]
)

inputs = keras.Input(shape=(180, 180, 3))
x = data_augmentation(inputs)
x = keras.layers.Lambda(
     lambda x: keras.applications.vgg16.preprocess_input(x))(x)
x = conv_base(x)
x = layers.Flatten()(x)
x = layers.Dense(256)(x)
x = layers.Dropout(0.5)(x)
outputs = layers.Dense(1, activation="sigmoid")(x)
model = keras.Model(inputs, outputs)
model.compile(loss="binary_crossentropy",
              optimizer="rmsprop",
              metrics=["accuracy"])
```

```
loss: 1.9956 - accuracy: 0.9770
```

b. Fine-tuning for above model with data augmentation

```
conv_base.trainable = True
for layer in conv_base.layers[:-4]:
    layer.trainable = False
```

```
model.compile(loss="binary_crossentropy",

optimizer=keras.optimizers.RMSprop(learning_rate=1e-5),
              metrics=["accuracy"])
```

Test accuracy: 0.9750, Test loss: 1.7709

Step 2: training set size 3000

a. Feature extraction with a pretrained model
with data augmentation and dropout layers.
`loss: 0.7631 - accuracy: 0.9830`

b. Fine-tuning for above model with data augmentation

Test accuracy: 0.9840, Test loss: 0.9174

Step 3: training set size 6000

a. Feature extraction with a pretrained model
with data augmentation and dropout layers.
`loss: 0.5502 - accuracy: 0.9790`

b. Fine-tuning for above model with data augmentation

Test accuracy: 0.9810, Test loss: 0.2363

Conclusion: Increasing the training set size for the pretrained model didn't make much difference; the improvement in test set accuracy was less than 1%.