```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=9473189

Enter your authorization code:
··········
Mounted at /content/drive
```

```
## Methods and Imports

import numpy as np
from math import log
import matplotlib.pyplot as plt
import cv2
import os
import time
import random
from sklearn.utils import shuffle

import tensorflow.compat.v1 as tf
tf.disable_v2_behavior()
from tensorflow.keras.layers import Flatten

# Read sample as either greyscale or color from path provided
def get_samples(path):
    samples = []
    for file in os.listdir(path):
        samples.append(cv2.imread(path + file, cv2.IMREAD_COLOR))
    return samples

# Increase size of image from 20x20 to 28x28 to match standard LeNet 5 artchitecture
def upscale(data):
    scale_percent = 140 # percent of original size
    image_list = []
    for img in data:
        dst = cv2.resize(img, None, fx = 1.4, fy = 1.4, interpolation = cv2.INTER_CUBIC)
        image_list.append(dst)
    return image_list

# Convert image array to grey scale
def grayscale(data):
    image_list = []
    for img in data:
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        image_list.append(gray)
    image_list = np.array(image_list)
    image_list = np.reshape(image_list, (-1,32,32,1))
    return np.array(image_list)
```

```
# Provided validation/testing split and positive/negative samples, create arrays for train/te
def data_prep(valid_split,testing_split,positive_samples,negative_samples):

    validation_split = valid_split * testing_split

    # Point of split for validation and test, for both positive and negative samples
    pos_split_test = int(len(positive_samples)*testing_split)
    pos_split_valid = int(len(positive_samples)*validation_split)
    neg_split_test = int(len(negative_samples)*testing_split)
    neg_split_valid = int(len(negative_samples)*validation_split)

    # Create the array to hold images in shape of [#samples, dim1, dim2, color channels]
    training_set = positive_samples[0:pos_split_valid] + negative_samples[0:neg_split_valid]
    validation_set = positive_samples[pos_split_valid:pos_split_test] + negative_samples[neg_
    testing_set = positive_samples[pos_split_test:] + negative_samples[neg_split_test:]

    # Add 0's and 1's for image labels
    training_labels = [1] * len(positive_samples[0:pos_split_valid]) + [0] * len(negative_sam
    validation_labels = [1] * len(positive_samples[pos_split_valid:pos_split_test]) + [0] * l
    testing_labels = [1] * len(positive_samples[pos_split_test:]) + [0] * len(negative_sample

    # Pad images with 0s
    training_set      = np.pad(training_set, ((0,0),(2,2),(2,2),(0,0)), 'constant')
    validation_set    = np.pad(validation_set, ((0,0),(2,2),(2,2),(0,0)), 'constant')
    testing_set       = np.pad(testing_set, ((0,0),(2,2),(2,2),(0,0)), 'constant')
    return training_set,training_labels,validation_set,validation_labels,testing_set,testing_


# build using tensorflow for LeNet 5 network
def LeNet(x, activ_func):
    mu = 0
    sigma = 0.1

    # convolutional layer 1. input = 32x32x1. output = 28x28x6
    conv1_W = tf.Variable(tf.truncated_normal(shape=(5,5,1,6), mean=mu, stddev=sigma))
    conv1_b = tf.Variable(tf.zeros(6))
    conv1 = tf.nn.conv2d(x, conv1_W, strides=[1,1,1,1], padding='VALID') + conv1_b

    # activation with relu
    conv1 = activ_func(conv1)

    # max pooling. input = 28x28x6. output = 14x14x6
    conv1 = tf.nn.max_pool(conv1, ksize=[1,2,2,1], strides=[1,2,2,1], padding='VALID')

    # convolutional layer 2. input = 14x14x6. output = 10x10x16
    conv2_W = tf.Variable(tf.truncated_normal(shape=(5,5,6,16), mean=mu, stddev=sigma))
    conv2_b = tf.Variable(tf.zeros(16))
    conv2 = tf.nn.conv2d(conv1, conv2_W, strides=[1,1,1,1], padding='VALID') + conv2_b

    # activation with relu
    conv2 = activ_func(conv2)
```

```
    # max pooling. input = 10x10x16. output = 5x5x16
    conv2 = tf.nn.max_pool(conv2, ksize=[1,2,2,1], strides=[1,2,2,1], padding='VALID')

    # flatten. input = 10x10x6. output = 400
    fc0   = Flatten()(conv2)

    # layer 3: fully connected layer. input = 400. output = 120
    fc1_W = tf.Variable(tf.truncated_normal(shape=(400,120), mean=mu, stddev=sigma))
    fc1_b = tf.Variable(tf.zeros(120))
    fc1 = tf.matmul(fc0, fc1_W) + fc1_b

    # activation with relu
    fc1 = activ_func(fc1)

    # drop out to prevent overfitting
    fc1 = tf.nn.dropout(fc1, keep_probability)

    # layer 4: fully connected layer. input = 120. output = 84
    fc2_W = tf.Variable(tf.truncated_normal(shape=(120,84), mean=mu, stddev=sigma))
    fc2_b = tf.Variable(tf.zeros(84))
    fc2 = tf.matmul(fc1, fc2_W) + fc2_b

    # activation with relu
    fc2 = activ_func(fc2)

    fc2 = tf.nn.dropout(fc2, keep_probability)

    # layer 5: fully connected layer. input = 84. output = 2
    fc3_W = tf.Variable(tf.truncated_normal(shape=(84,2), mean=mu, stddev=sigma))
    fc3_b = tf.Variable(tf.zeros(2))
    logits = tf.matmul(fc2, fc3_W) + fc3_b

    return logits

def evaluate(X_data, y_data):
    num_examples = len(X_data)
    total_accuracy = 0
    sess = tf.get_default_session()
    for offset in range(0, num_examples, BATCH_SIZE):
        batch_x, batch_y = X_data[offset:offset+BATCH_SIZE], y_data[offset:offset+BATCH_SIZE]
        accuracy = sess.run(accuracy_operation, feed_dict={x: batch_x, y: batch_y, keep_proba
        total_accuracy += (accuracy * len(batch_x))
    return total_accuracy / num_examples
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/compat/
Instructions for updating:
non-resource variables are not supported in the long term
```

```
# Generate samples from Dataset
```

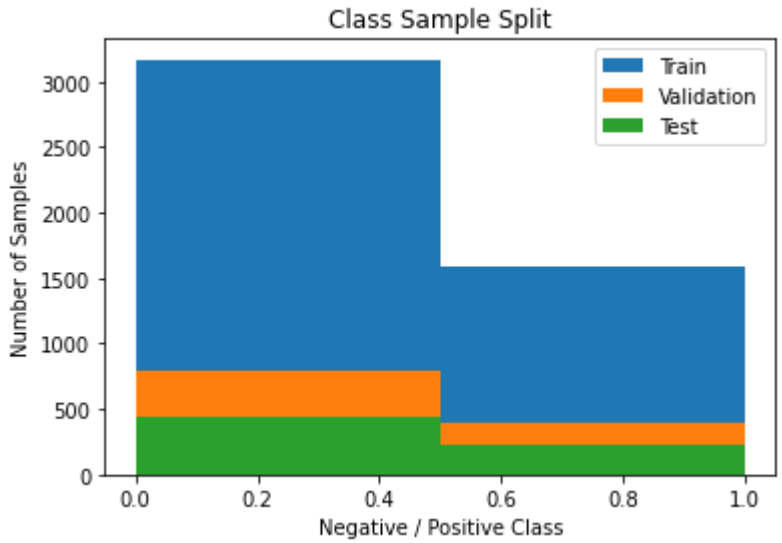
```
positive_samples = get_samples("/content/drive/My Drive/school/Machine Learning/ECE 763/posit
negative_samples = get_samples("/content/drive/My Drive/school/Machine Learning/ECE 763/negat
print('Original Image Dimensions : ',positive_samples[0].shape)

# Upscale dimensions
positive_samples = upscale(positive_samples)
negative_samples = upscale(negative_samples)
print('Rezied Image Dimensions : ',positive_samples[0].shape)
print()

print("Total samples before Data Augmentation: ")
print("Positives: " + str(len(positive_samples)))
print("Negatives: " + str(len(negative_samples)))

# Data Augmentation - rotate negative images
negative_samples_90 = [np.rot90(el) for el in negative_samples]
negative_samples_180 = [np.rot90(el, 2) for el in negative_samples]
negative_samples_270 = [np.rot90(el, 3) for el in negative_samples]
negative_samples = negative_samples + negative_samples_90 + negative_samples_180 + negative_s

# Data Augmentation - horizontally flip positive images
positive_samples_flip = [np.flip(el, 1) for el in positive_samples]
positive_samples = positive_samples + positive_samples_flip

print("Total samples after Data Augmentation: ")
print("Positives: " + str(len(positive_samples)))
print("Negatives: " + str(len(negative_samples)))
```

```
Original Image Dimensions :  (20, 20, 3)
Rezied Image Dimensions :  (28, 28, 3)

Total samples before Data Augmentation:
Positives: 1100
Negatives: 1100
Total samples after Data Augmentation:
Positives: 2200
Negatives: 4400
```

```
# Generate arrays of labels and images for split data
X_train,y_train,X_valid,y_valid,X_test,y_test = data_prep(0.80, 0.90,positive_samples,negativ

# Print info on split image arrays
n_classes = np.unique(y_train).size
print("Image Shape after padding: {}".format(X_train[0].shape))
print("Number of Classes: {}".format(n_classes))
print("Training Set:   {} samples".format(len(X_train)))
print("Validation Set: {} samples".format(len(X_valid)))
print("Test Set:       {} samples".format(len(X_test)))
```

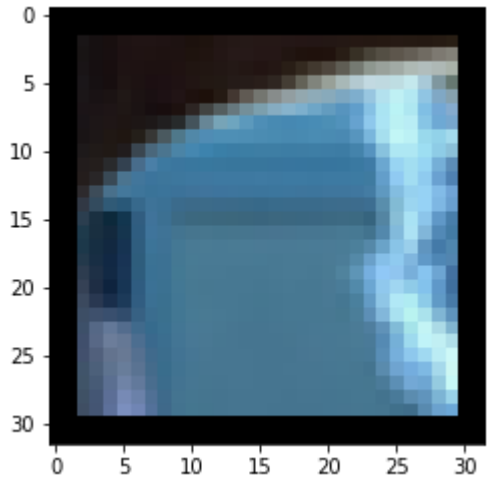
```
Image Shape after padding: (32, 32, 3)
Number of Classes: 2
Training Set:   4752 samples
Validation Set: 1188 samples
Test Set:       660 samples
```

```
# Distribution of labels
plt.hist(y_train, bins=n_classes, label='Train')
plt.hist(y_valid, bins=n_classes, label='Validation')
plt.hist(y_test, bins=n_classes, label='Test')
plt.tick_params(axis='x')
plt.tick_params(axis='y')
plt.ylabel("Number of Samples")
plt.xlabel("Negative / Positive Class")
plt.legend()
plt.title("Class Sample Split")
plt.show()
```



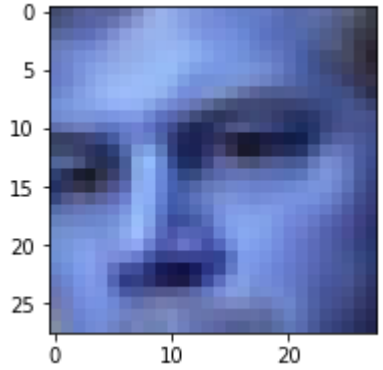

```
# Print an example of image

index = random.randint(0, len(X_train))
image = X_train[index]

print('Image label:' + str(y_train[index]))
plt.imshow(image)
```

```
Image label:0
<matplotlib.image.AxesImage at 0x7f594f2254e0>
```

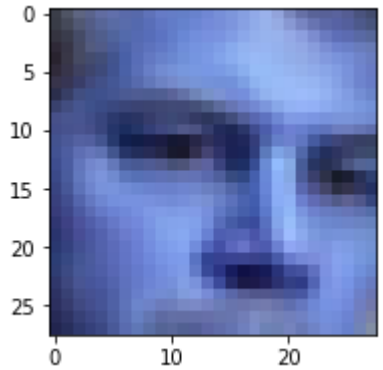



```
# Printing Data Augmentation


for test_var in [positive_samples,positive_samples_flip,negative_samples,negative_samples_90,
  image = test_var[0]
  plt.figure()
  plt.figure(figsize=(3,3))
  plt.imshow(image)
```

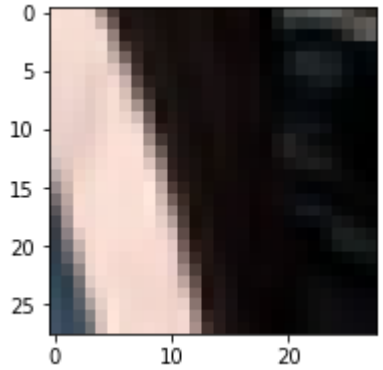
```
<Figure size 432x288 with 0 Axes>
```



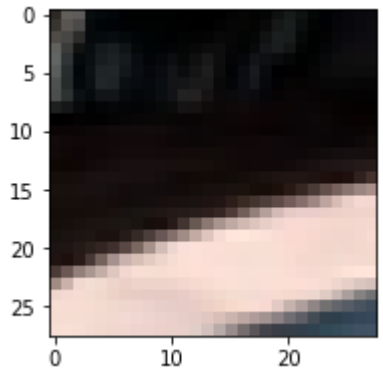
```
<Figure size 432x288 with 0 Axes>
```



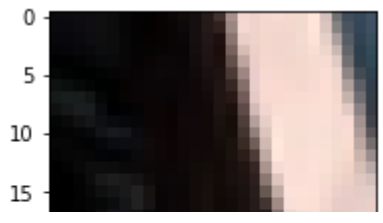
```
<Figure size 432x288 with 0 Axes>
```

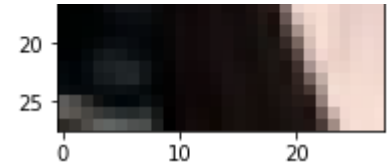


```
<Figure size 432x288 with 0 Axes>
```

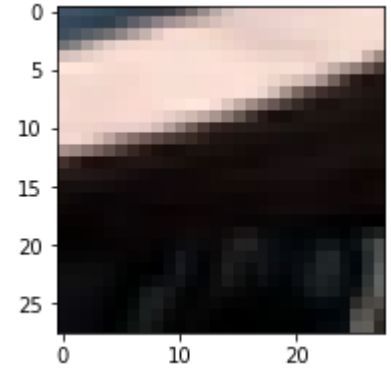


```
<Figure size 432x288 with 0 Axes>
```



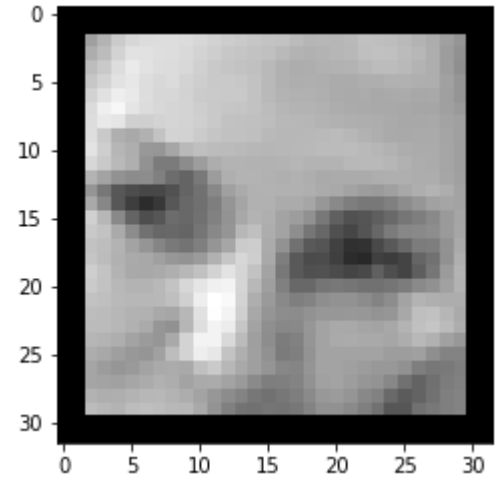


```
<Figure size 432x288 with 0 Axes>
```



```
# grascale data - CAN'T RERUN
X_train = grayscale(X_train).astype('float')
X_valid = grayscale(X_valid).astype('float')
X_test = grayscale(X_test).astype('float')


# grayscale data - example
index = random.randint(0, len(X_train))
image = X_train[index].squeeze()

plt.imshow(image, cmap="gray")
print('Image Label (greyscale): ' + str(y_train[index]))
```

```
Image Label (greyscale): 1
```



```
# normalize data
X_train = (X_train - 128)/128
X_valid = (X_valid - 128)/128
X_test = (X_test - 128)/128

# shuffle data
```

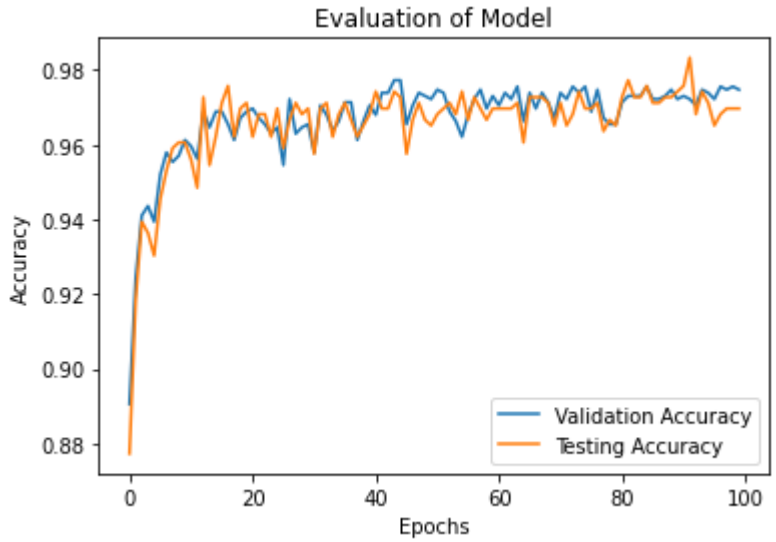
```
    # shuffle data
    X_train, y_train = shuffle(X_train, y_train)
    X_valid, y_valid = shuffle(X_valid, y_valid)
    X_test, y_test = shuffle(X_test, y_test)


    ## Training and Evaluation Pipeline

    best_valacc = np.zeros((100,), dtype=int)
    best_testacc = np.zeros((100,), dtype=int)
    # Best: [rate_var, dr_var, batch_var, activ_var]
    best_model = [0,0,0,0]

    for rate_var in [.001, .002, .003]:
      for dr_var in [.1, .2, .3]:
        for batch_var in [64, 128, 256]:
          for activ_var in [tf.nn.relu,tf.nn.relu6,tf.nn.selu]:

            # Save accuracy metrics for plotting and evaluation
            save_valacc = []
            save_testacc = []

            # Hyperparameter testing

            #[.001, .002, .003]
            rate = rate_var
            #[.1, .2, .3]
            dropout_rate = dr_var
            #[64, 128, 256]
            BATCH_SIZE = batch_var
            #[tf.nn.relu,tf.nn.relu6,tf.nn.selu]
            activ_func = activ_var


            # Regression Analysis
            #['LASSO','RIDGE','NONE']
            regression_type = 'RIDGE'

            # Set high to measure overfitting
            EPOCHS = 100

            # Features and Labels for evaluation
            x = tf.placeholder(tf.float32, (None, 32, 32, 1))
            y = tf.placeholder(tf.int32, (None))
            keep_probability = tf.placeholder(tf.float32)
            one_hot_y = tf.one_hot(y, 2)

            # Create variables for linear regression
            A = tf.Variable(tf.random_normal(shape=[1,1]))
            b = tf.Variable(tf.random_normal(shape=[1,1]))

            # Initialize model and entropy function
            logits = LeNet(x, activ_func)
```

```
    logits = LeNet(x, dtrop_rate)
    cross_entropy = tf.nn.sigmoid_cross_entropy_with_logits(logits=logits, labels=one_hot

    # Select Regression Type
    if regression_type == 'LASSO':
        # Declare Lasso loss function
        # Lasso Loss = L2_Loss + heavyside_step,
        # Where heavyside_step ~ 0 if A < constant, otherwise ~ 99
        lasso_param = tf.constant(0.9)
        heavyside_step = tf.truediv(1., tf.add(1., tf.exp(tf.multiply(-50., tf.subtract(A
        regularization_param = tf.multiply(heavyside_step, 99.)
        loss_operation = tf.add(tf.reduce_mean(cross_entropy), regularization_param)

    elif regression_type == 'RIDGE':
        # Declare the Ridge loss function
        # Ridge loss = L2_loss + L2 norm of slope
        ridge_param = tf.constant(1.)
        ridge_loss = tf.reduce_mean(tf.square(A))
        loss_operation = tf.expand_dims(tf.add(tf.reduce_mean(cross_entropy), tf.multiply

    else:
        loss_operation = tf.reduce_mean(cross_entropy)

    optimizer = tf.train.AdamOptimizer(learning_rate = rate)
    training_operation = optimizer.minimize(loss_operation)

    correct_prediction = tf.equal(tf.argmax(logits, 1), tf.argmax(one_hot_y, 1))
    accuracy_operation = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))

    saver = tf.train.Saver()

    with tf.Session() as sess:
        sess.run(tf.global_variables_initializer())
        num_examples = X_train.shape[0]
        print("Training for:")
        print("Learning rate:" + str(rate_var))
        print("Dropout rate: " + str(dr_var))
        print("Batch size: " + str(batch_var))
        print("Activation function: " + str(activ_var))
        print()
        for i in range(EPOCHS):
            X_train, y_train = shuffle(X_train, y_train)
            for offset in range(0, num_examples, BATCH_SIZE):
                end = offset + BATCH_SIZE
                batch_x, batch_y = X_train[offset:end], y_train[offset:end]
                sess.run(training_operation, feed_dict={x: batch_x, y: batch_y, keep_prob

            validation_accuracy = evaluate(X_valid, y_valid)
            test_accuracy = evaluate(X_test, y_test)

            save_valacc.append(validation_accuracy)
            save_testacc.append(test_accuracy)
```

```
        saver.save(sess, './lenet')
        print("Test average for last 50 epochs: " + str(np.mean(save_testacc[50:100])))
        print("Model saved")
        print()

    with tf.Session() as sess:
        saver.restore(sess, tf.train.latest_checkpoint('.'))

        test_accuracy = evaluate(X_test, y_test)
        print("Final Test Accuracy = {:.3f}".format(test_accuracy))

    # Print graph for every run
    t = np.arange(0, len(save_testacc), 1)
    line1 = plt.plot(t, save_valacc, label='Validation Accuracy')
    line2 = plt.plot(t, save_testacc, label='Testing Accuracy')
    plt.tick_params(axis='x')
    plt.tick_params(axis='y')
    plt.legend(loc='lower right')
    plt.ylabel("Accuracy")
    plt.xlabel("Epochs")
    plt.title("Evaluation of Model")
    plt.show()

    # Save best model for plotting
    if np.mean(save_testacc[50:100]) > np.mean(best_testacc[50:100]):
      best_model = [rate_var, dr_var, batch_var, activ_var]
      best_testacc = save_testacc
      best_valacc = save_valacc
```

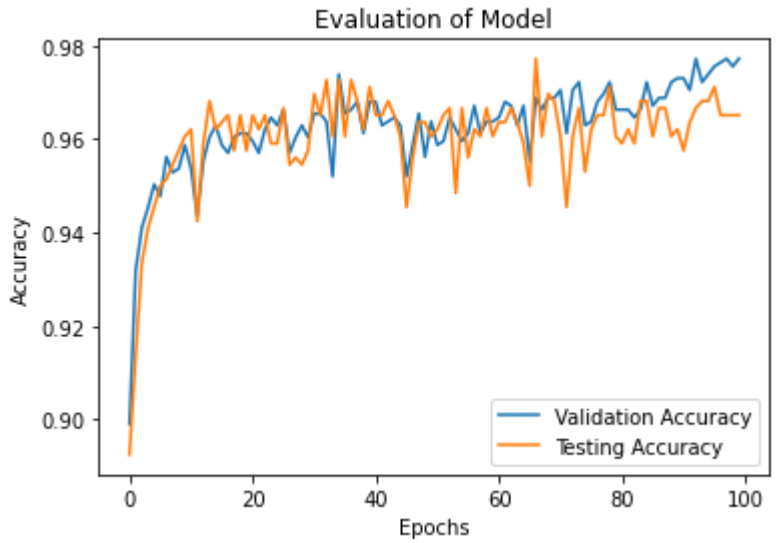
```
Training for:
Learning rate:0.001
Dropout rate: 0.1
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9693636359373727
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```


```
Training for:
Learning rate:0.001
Dropout rate: 0.1
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9705151509588416
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.970
```

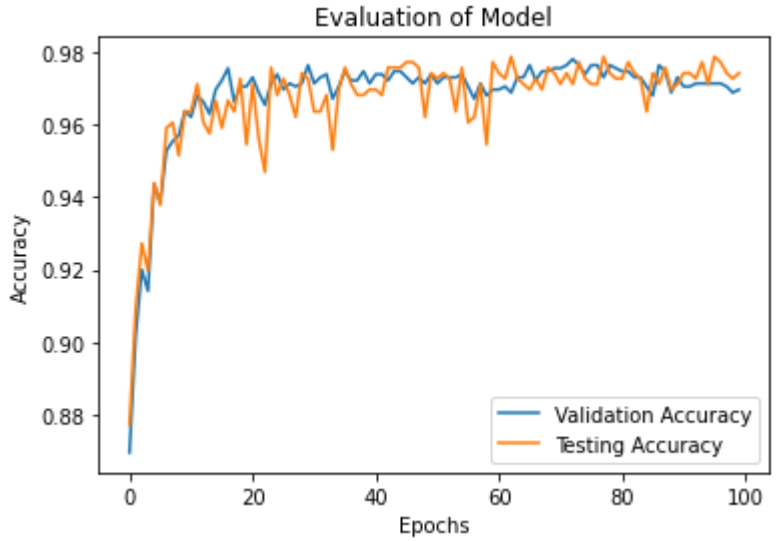



```
Training for:
Learning rate:0.001
Dropout rate: 0.1
```

```
Dropout rate: 0.1
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9631515152526623
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.965
```

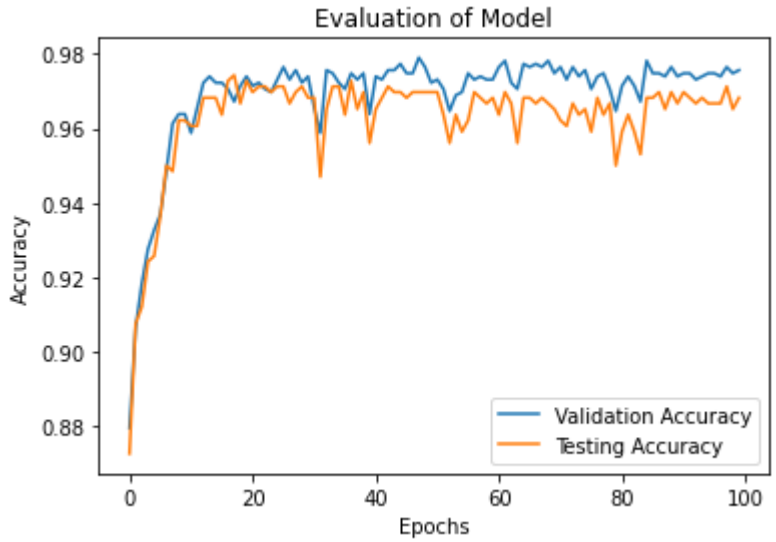



```
Training for:
Learning rate:0.001
Dropout rate: 0.1
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9724242420196534
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```

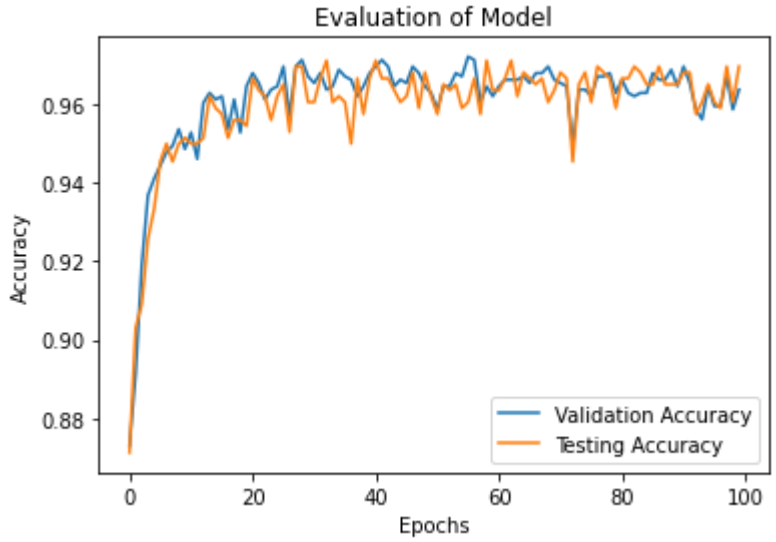



```
Training for:
Learning rate:0.001
Dropout rate: 0.1
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>
```

```
Test average for last 50 epochs: 0.9650303025245666
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.968
```

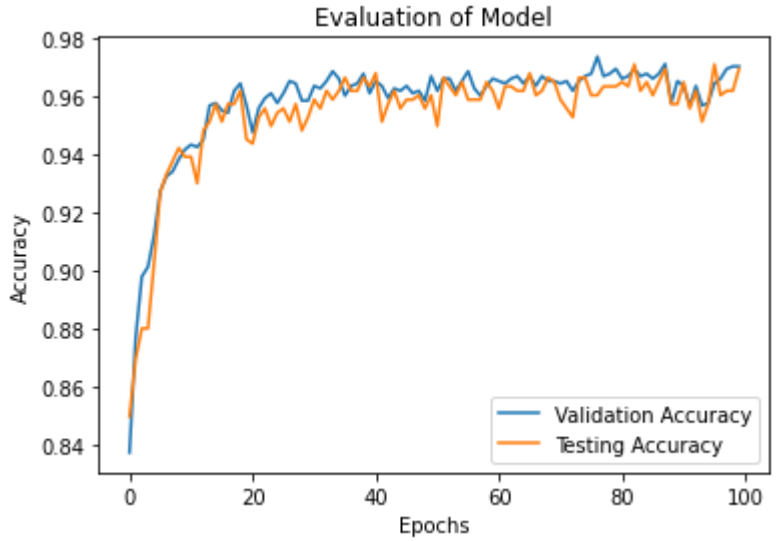



```
Training for:
Learning rate:0.001
Dropout rate: 0.1
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9645757572867655
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.970
```

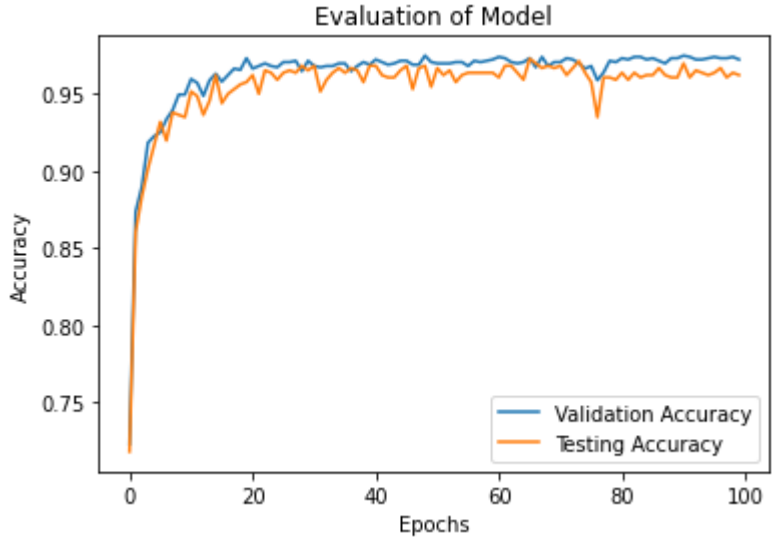



```
Training for:
Learning rate:0.001
Dropout rate: 0.1
Batch size: 256
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9622121202223228
Model saved
```

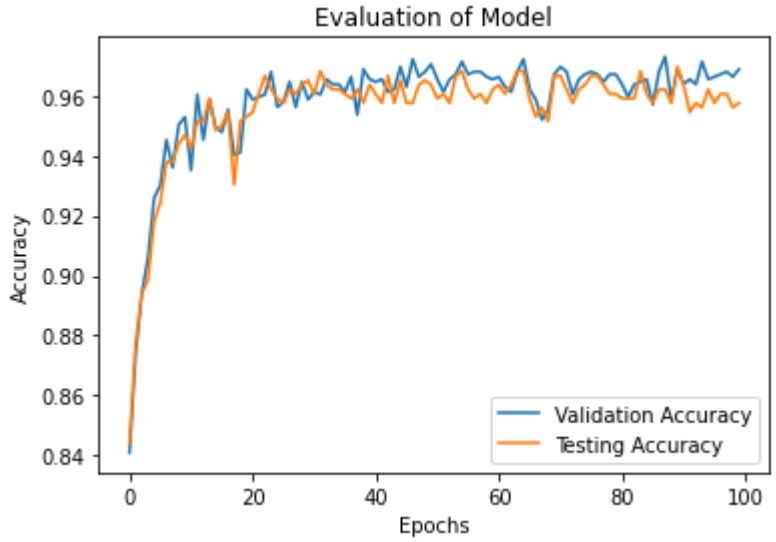
```
INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.970
```



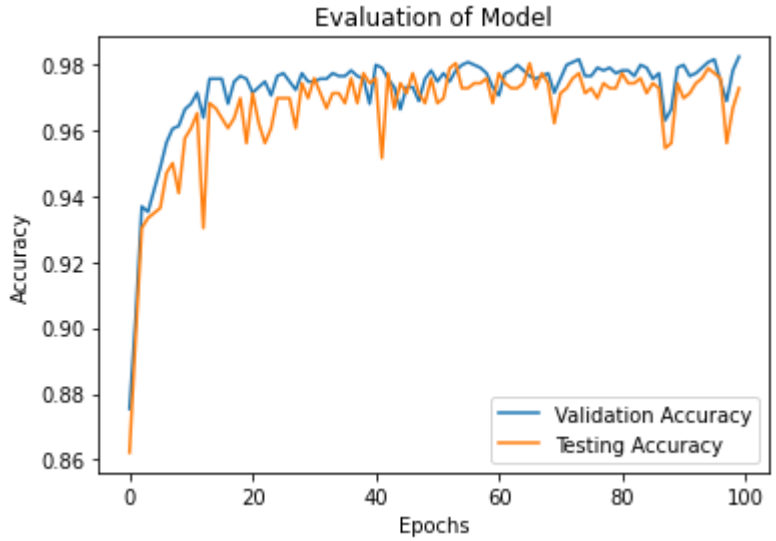

```
Training for:
Learning rate:0.001
Dropout rate: 0.1
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9631818162311206
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.962
```



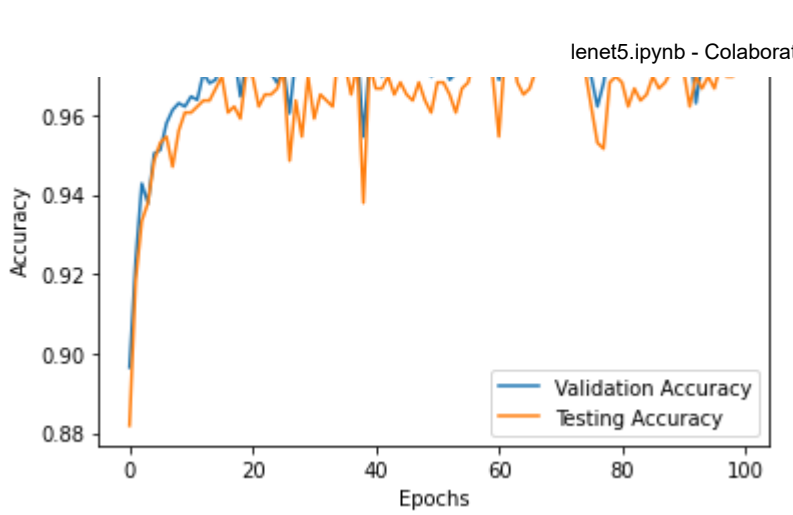

```
Training for:
Learning rate:0.001
Dropout rate: 0.1
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9611212111964371
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.958
```



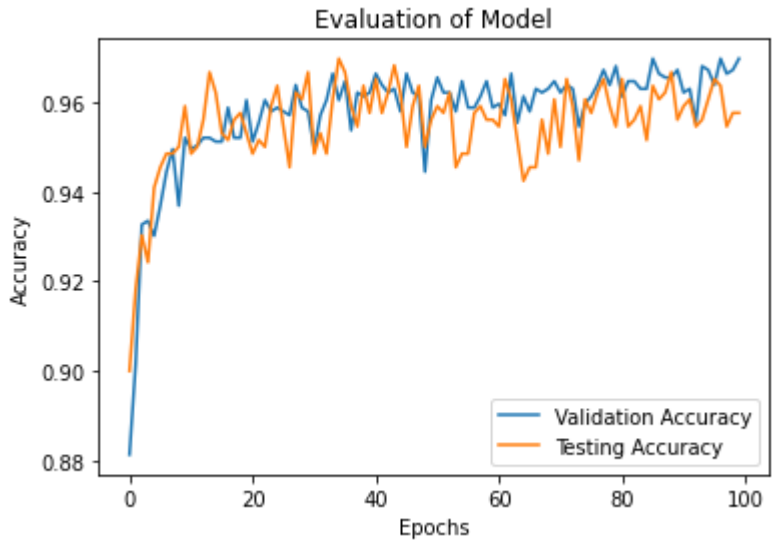

```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.972515151161136
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.973
```



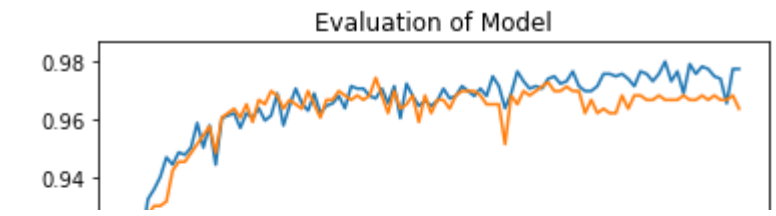

```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9686666660308838
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.976
```

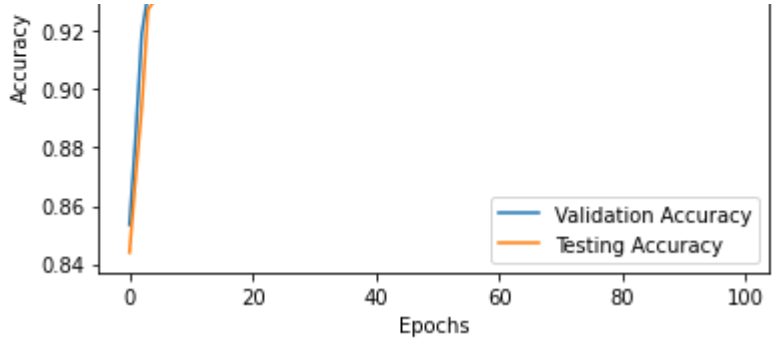



```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9569090905767497
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.958
```

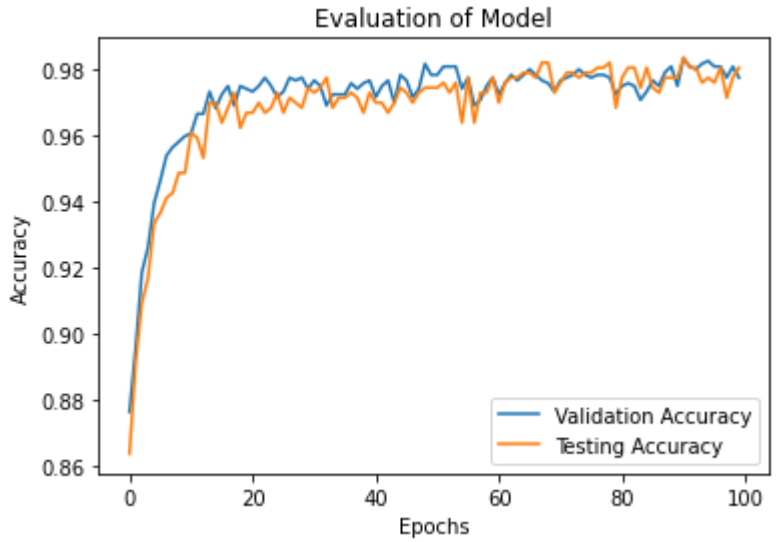



```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.966848484956857
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.964
```

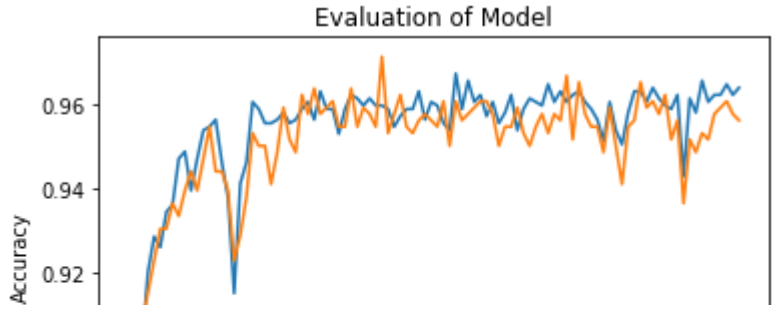


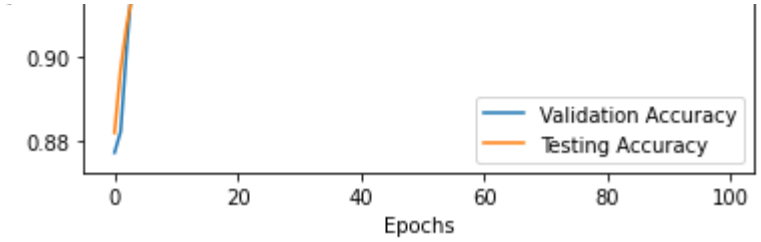



```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9765757576263312
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.980
```

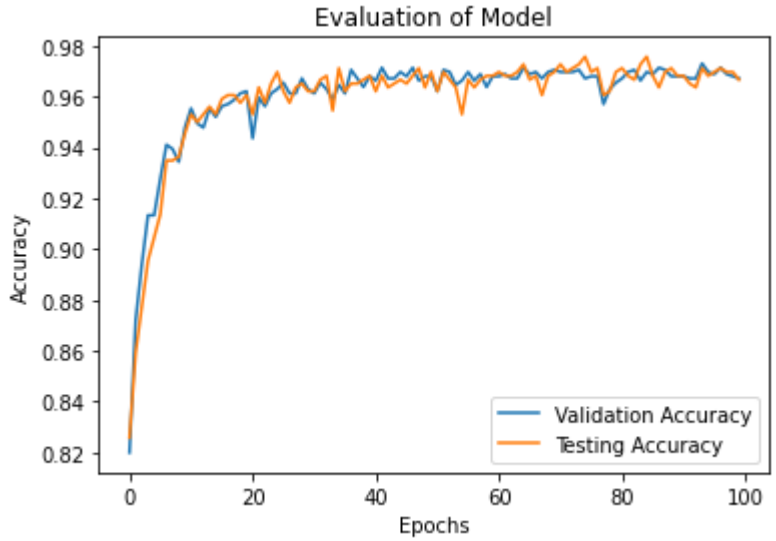



```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9556060604109909
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.956
```

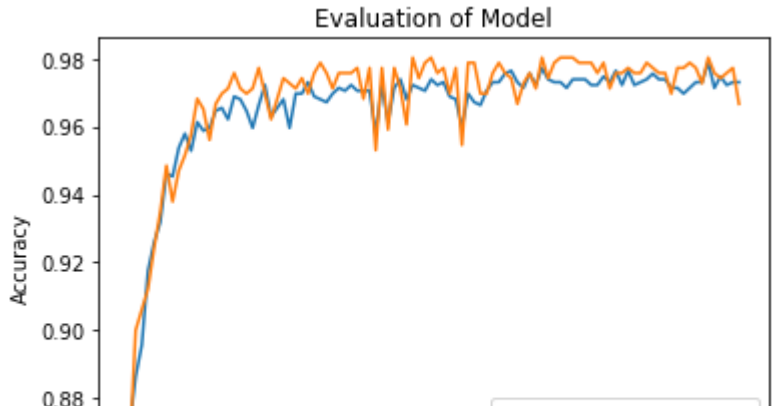


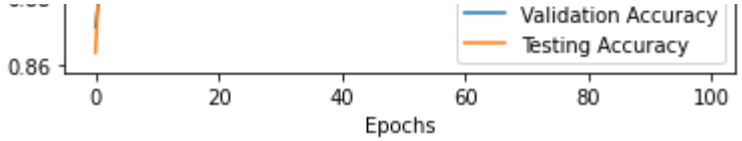



```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 256
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9680909096616687
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.967
```

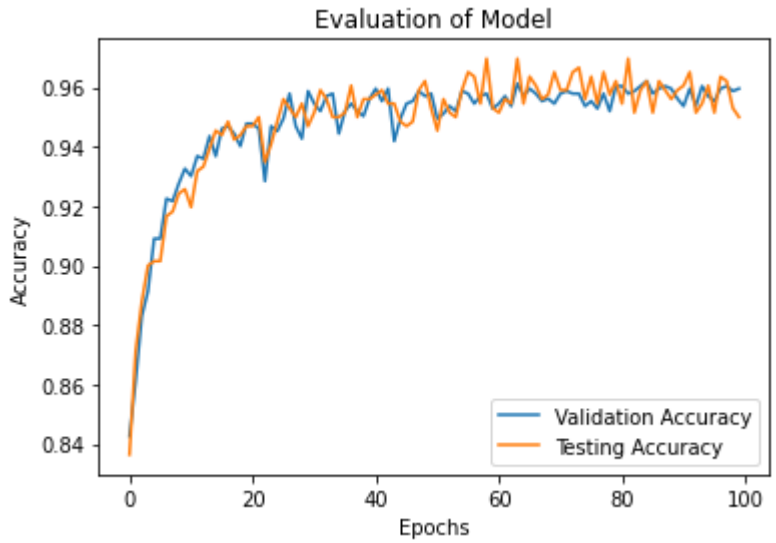



```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9753939408605748
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.967
```

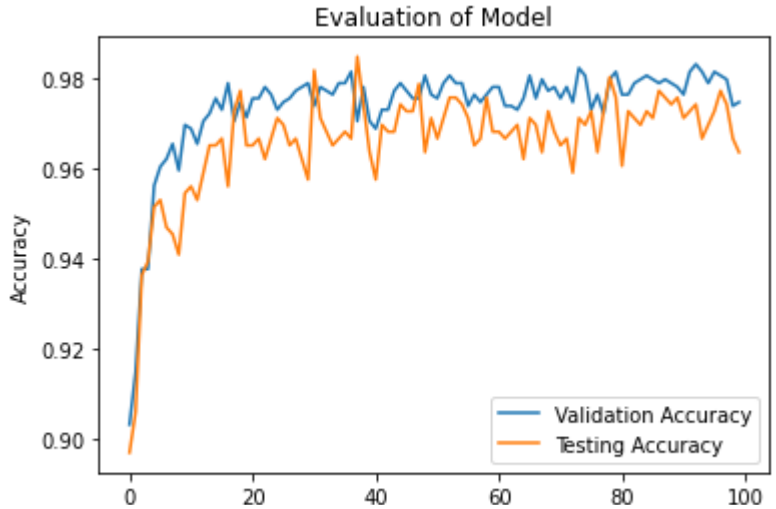




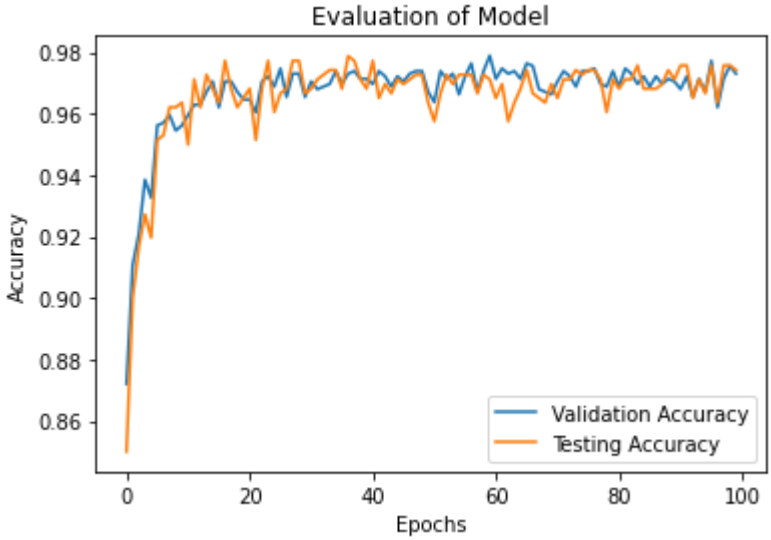

```
Training for:
Learning rate:0.001
Dropout rate: 0.2
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9584848478996392
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.950
```



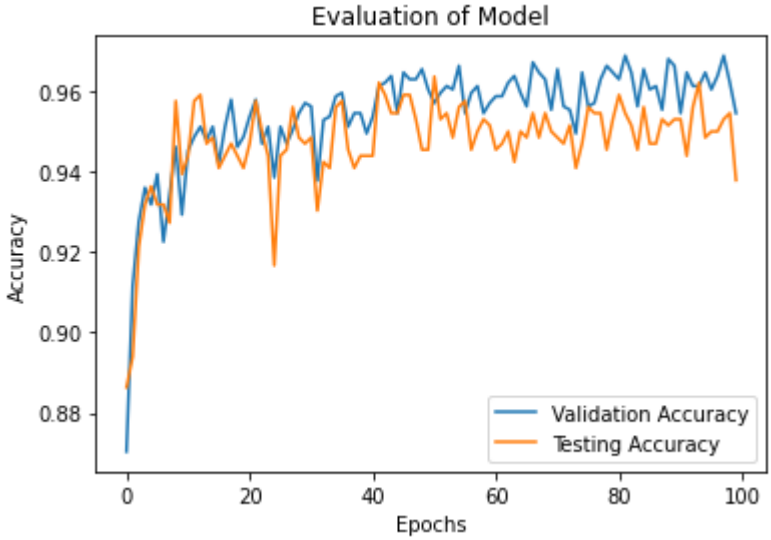

```
Training for:
Learning rate:0.001
Dropout rate: 0.3
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9703636357784271
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.964
```




```
Training for:
Learning rate:0.001
Dropout rate: 0.3
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.969818181406368
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```

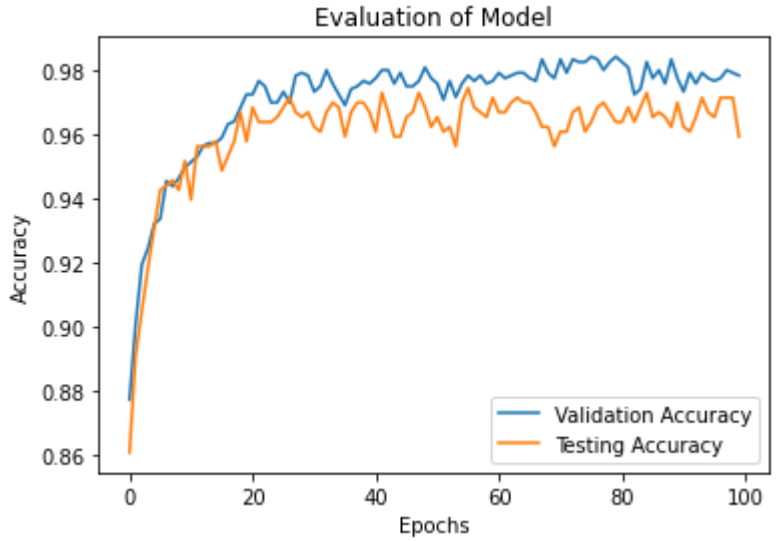



```
Training for:
Learning rate:0.001
Dropout rate: 0.3
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9509999999783255
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.938
```

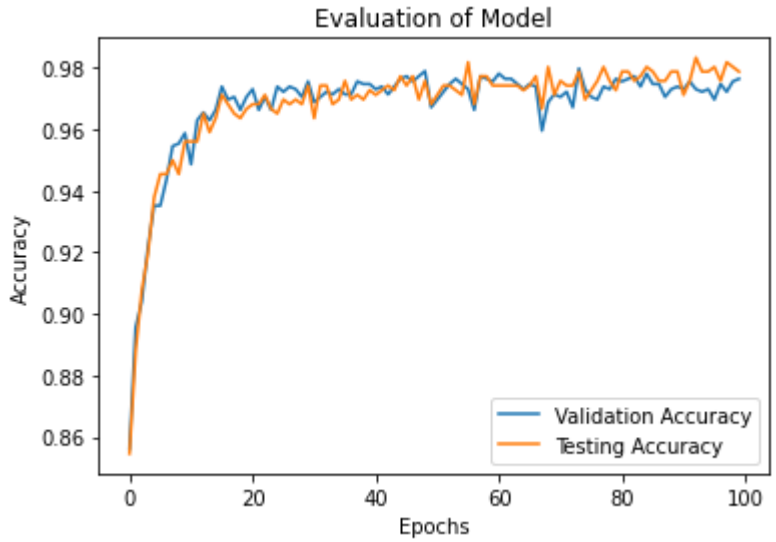



```
Training for:
Learning rate:0.001
```

```
Learning rate:0.001
Dropout rate: 0.3
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9659393940766653
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.959
```

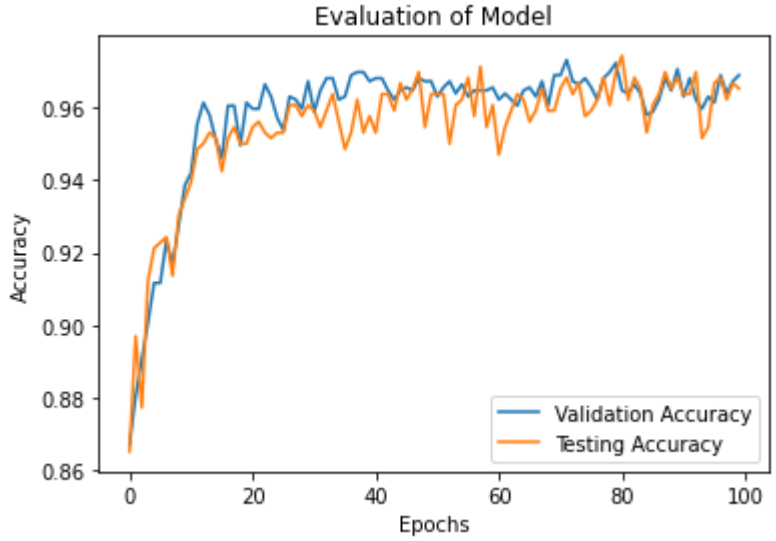



```
Training for:
Learning rate:0.001
Dropout rate: 0.3
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9759090914076025
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.979
```

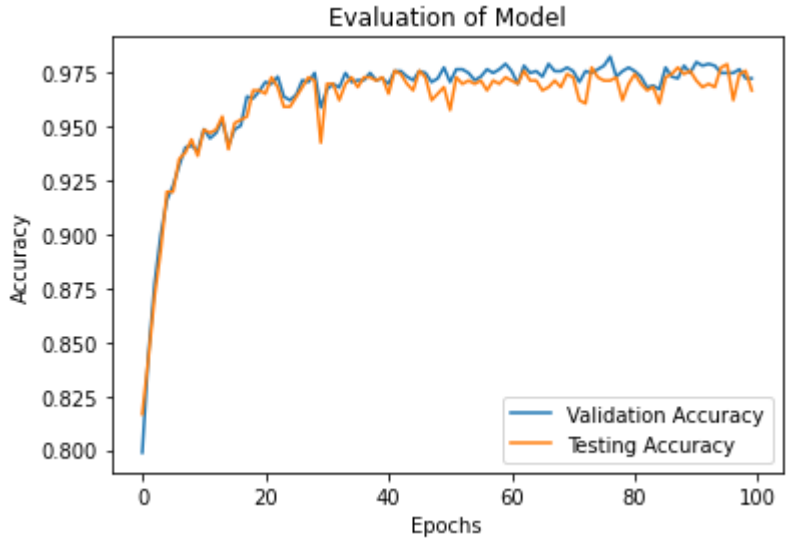



```
Training for:
Learning rate:0.001
Dropout rate: 0.3
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>
```

```
Activation function: <function selu at 0x7f59ec8001e0>

Test average for last 50 epochs: 0.9623939391338463
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.965
```



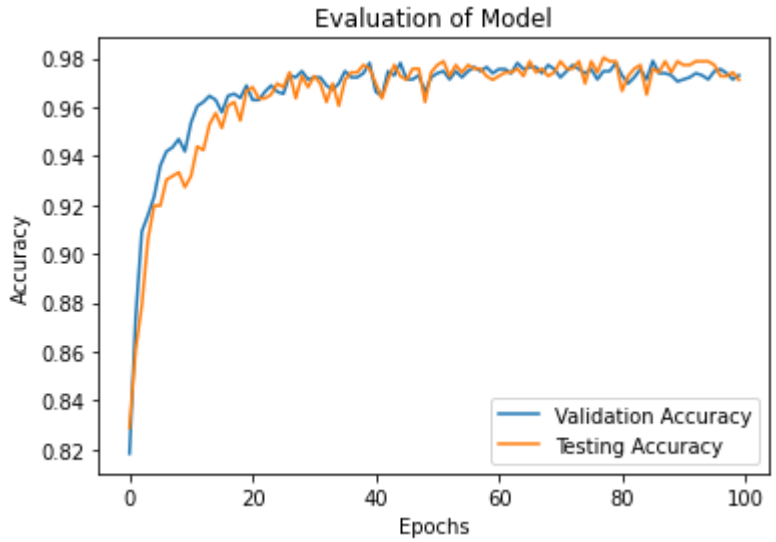

```
Training for:
Learning rate:0.001
Dropout rate: 0.3
Batch size: 256
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9703636359590473
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.967
```



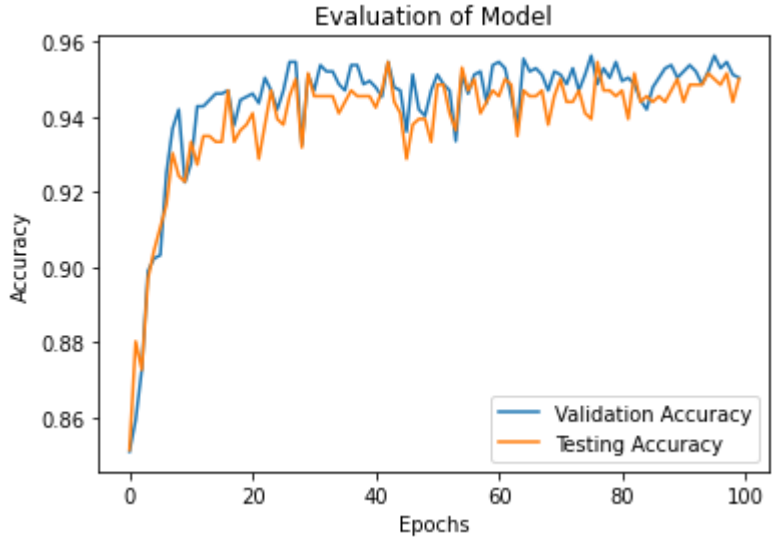

```
Training for:
Learning rate:0.001
Dropout rate: 0.3
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9753333329359691
Model saved
```

Model saved

```
INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```

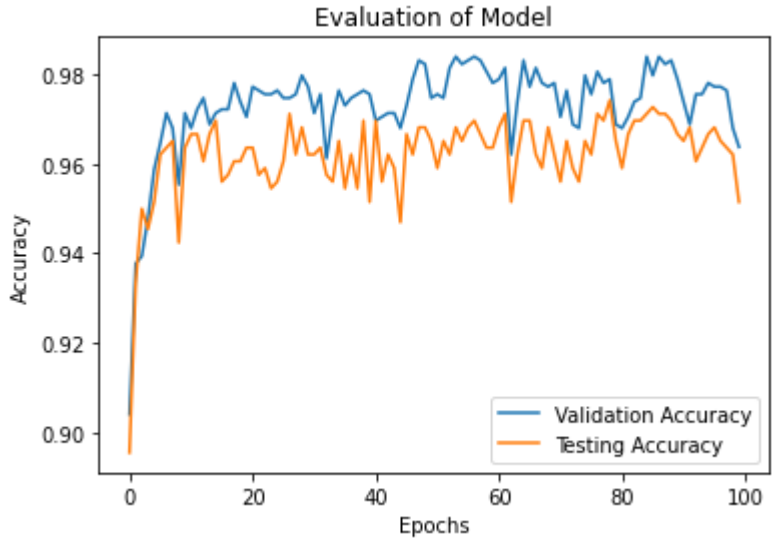



```
Training for:
Learning rate:0.001
Dropout rate: 0.3
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.946090908722444
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.950
```

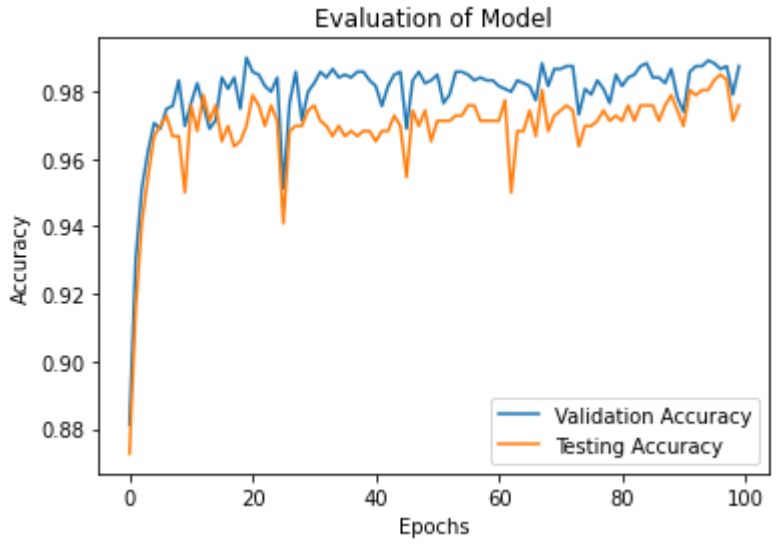



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9652424242640987
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.952
```

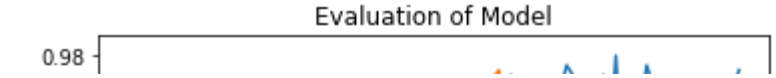
Final Test Accuracy = 0.952

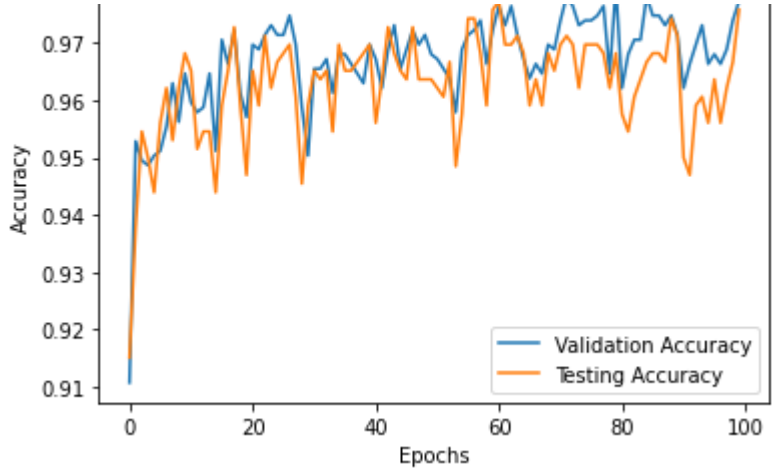



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9734242419835293
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.976
```

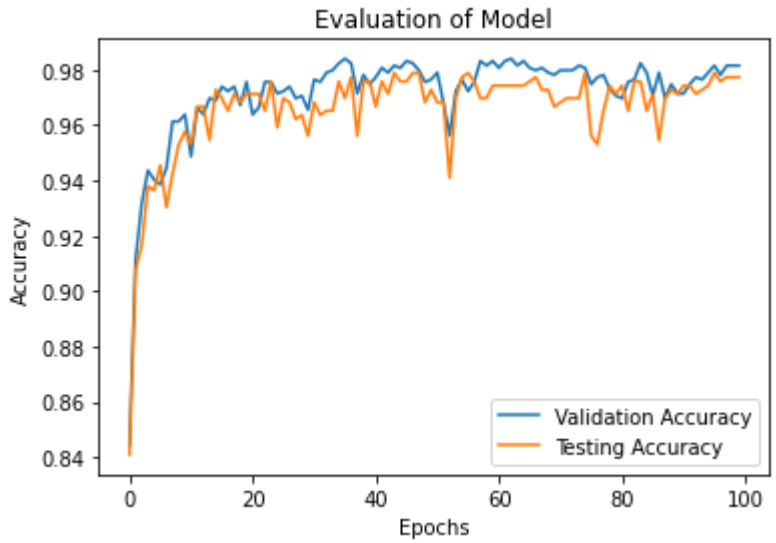



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9649393942428358
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.976
```

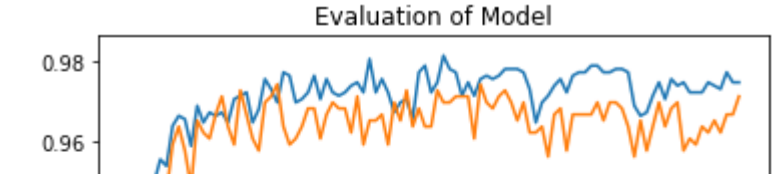


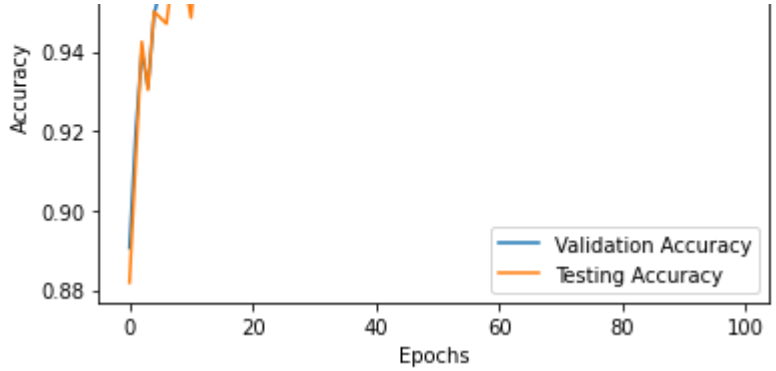



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9710606057571641
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.977
```

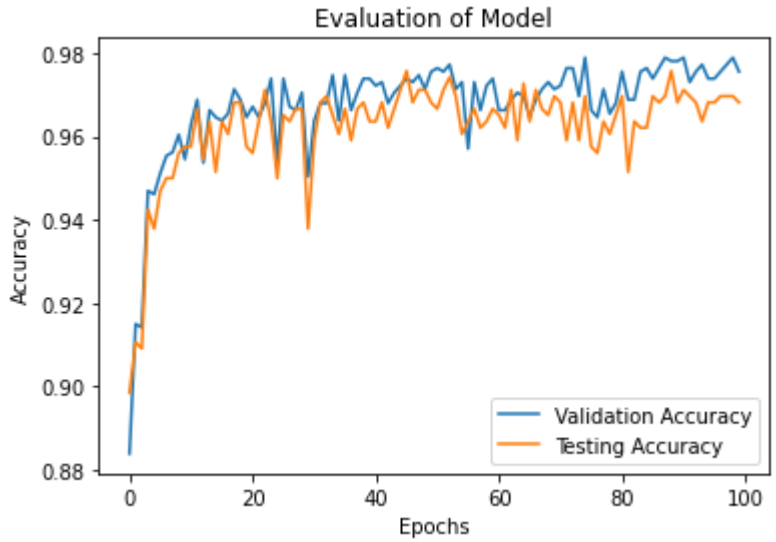



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.966090909466599
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```

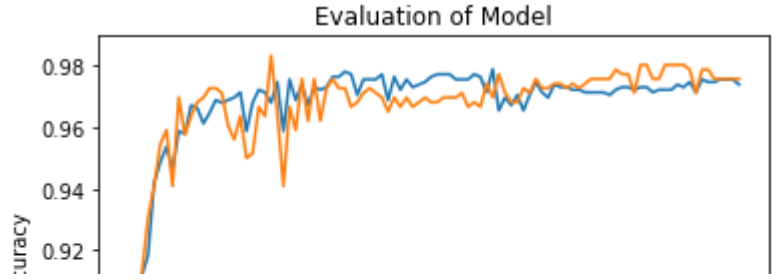


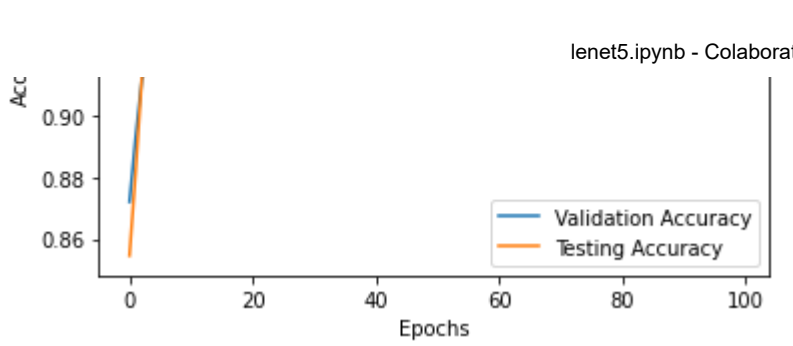



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9661212127425454
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.968
```

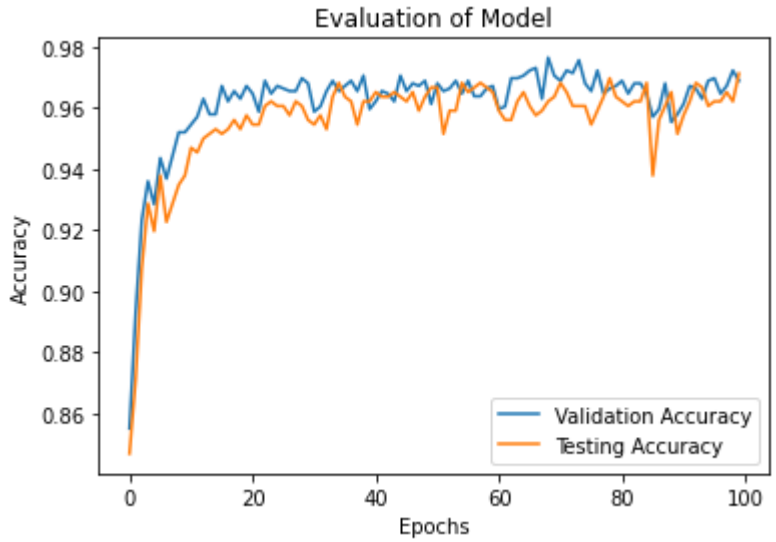



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 256
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9742727280168821
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.976
```

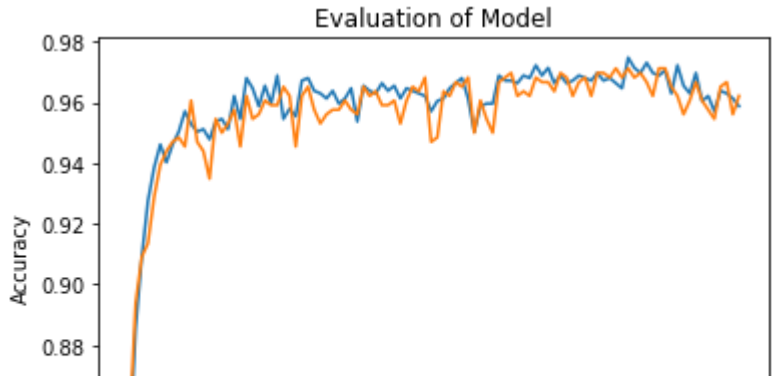


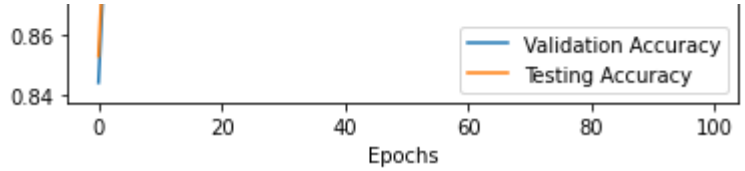



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9617575759165216
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```

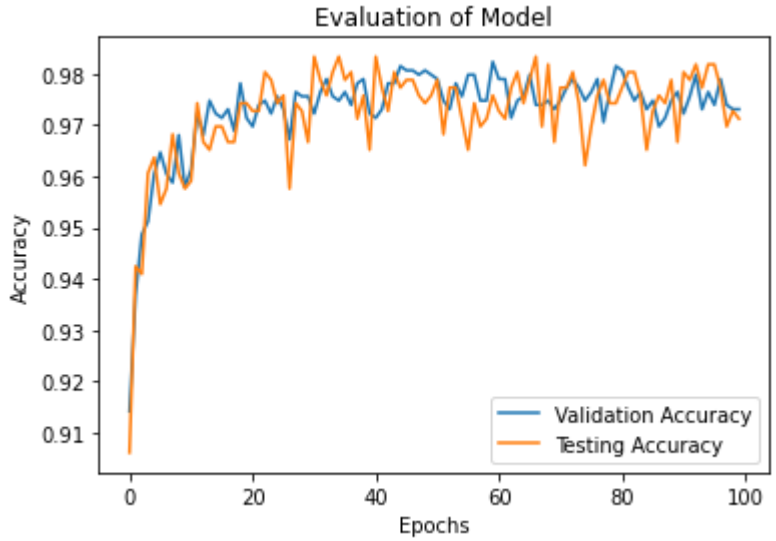



```
Training for:
Learning rate:0.002
Dropout rate: 0.1
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9640909088886144
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.962
```

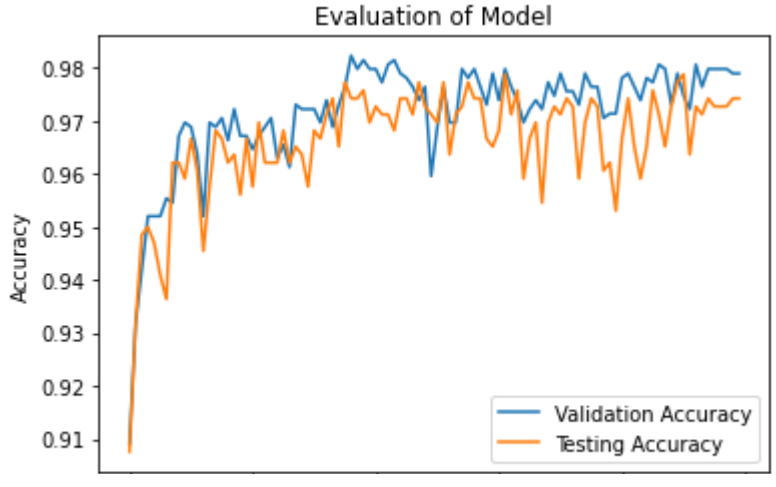


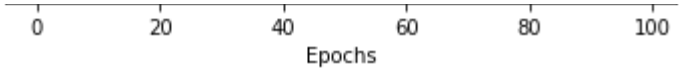



```
Training for:
Learning rate:0.002
Dropout rate: 0.2
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9749696966879295
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```

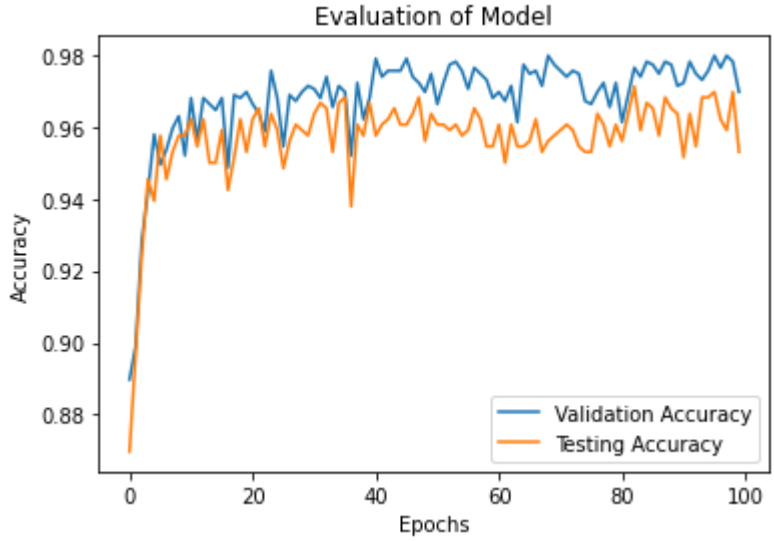



```
Training for:
Learning rate:0.002
Dropout rate: 0.2
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9697575753529867
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```





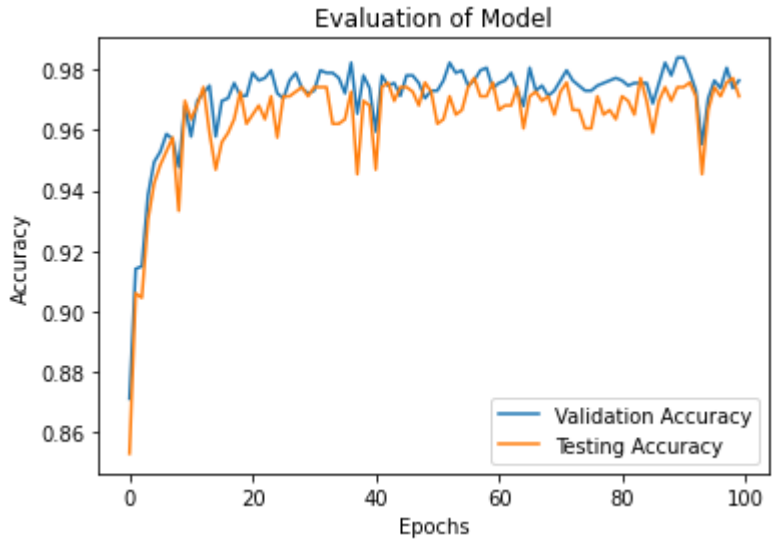

```
Training for:
Learning rate:0.002
Dropout rate: 0.2
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9598787874886483
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.953
```



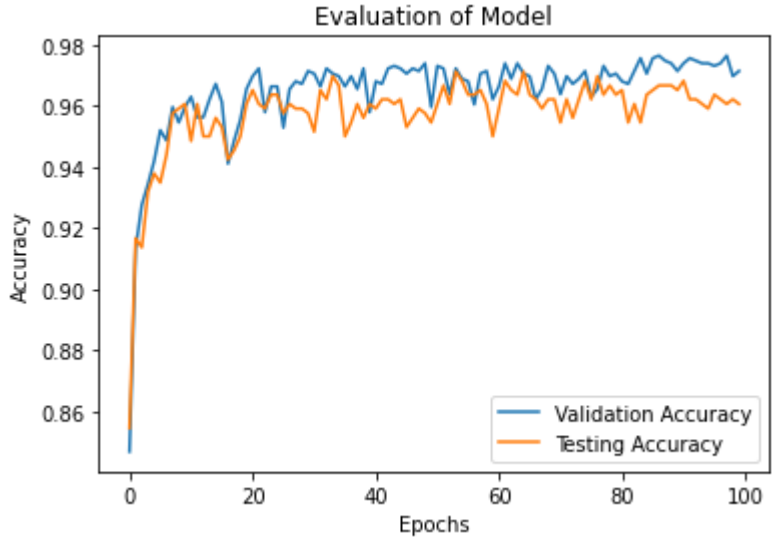

```
Training for:
Learning rate:0.002
Dropout rate: 0.2
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.971030302582365
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```


```
Training for:
```

```
Learning rate:0.002
Dropout rate: 0.2
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9691818179506244
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```

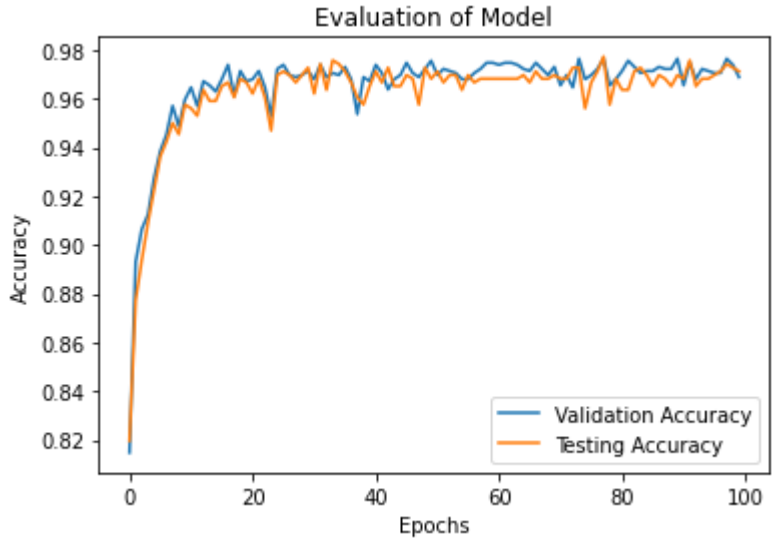



```
Training for:
Learning rate:0.002
Dropout rate: 0.2
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9629090914003777
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.961
```

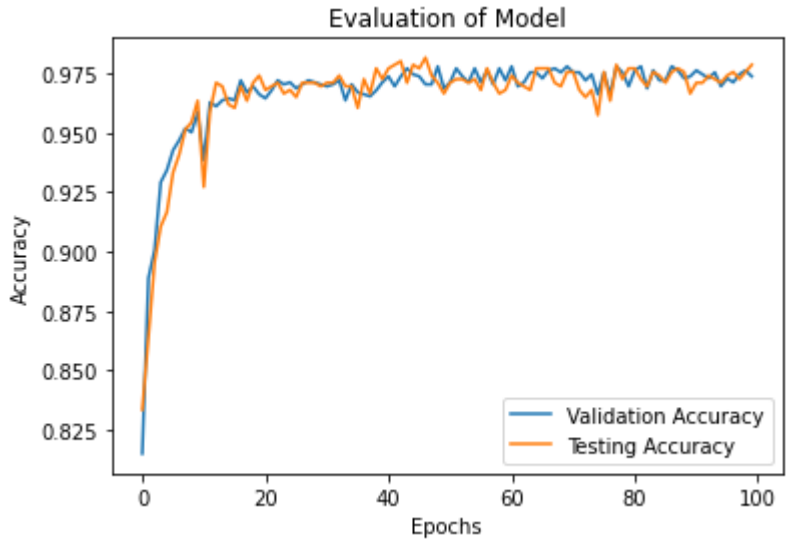



```
Training for:
Learning rate:0.002
Dropout rate: 0.2
Batch size: 256
```

```
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9686060583880454
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```



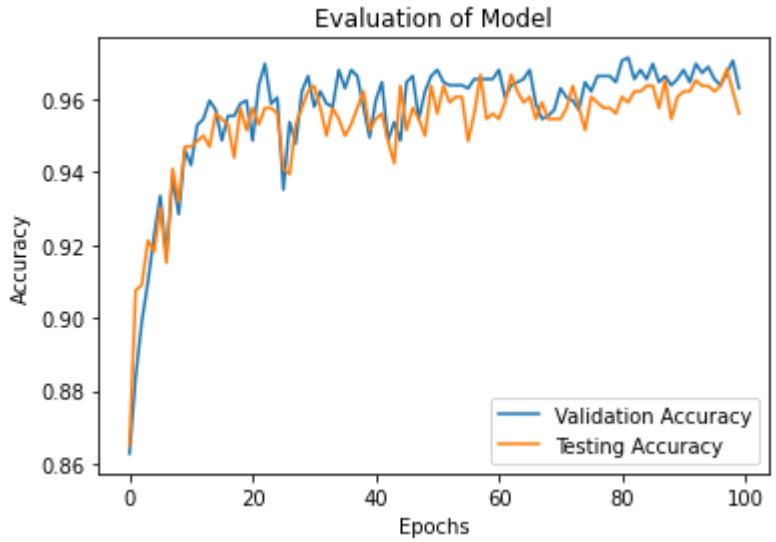

```
Training for:
Learning rate:0.002
Dropout rate: 0.2
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9724242434790641
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.979
```




```
Training for:
Learning rate:0.002
Dropout rate: 0.2
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9596060603026189
```

Model saved

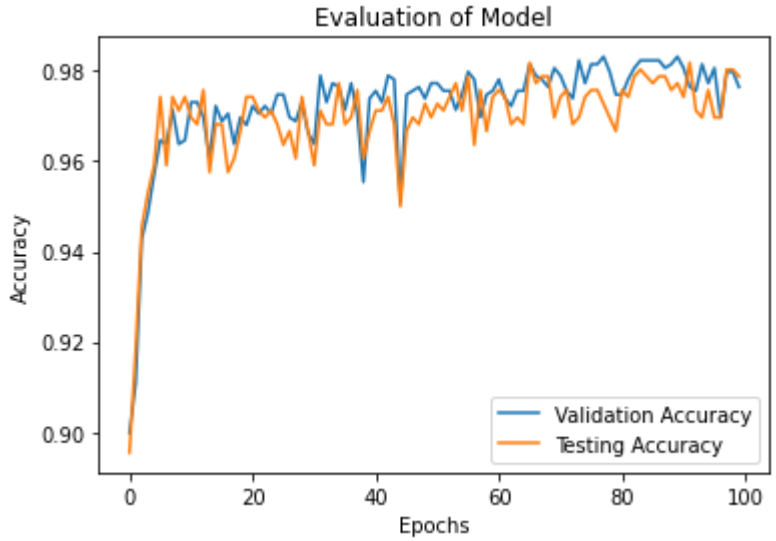
INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.956



Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9743636358073263
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.979

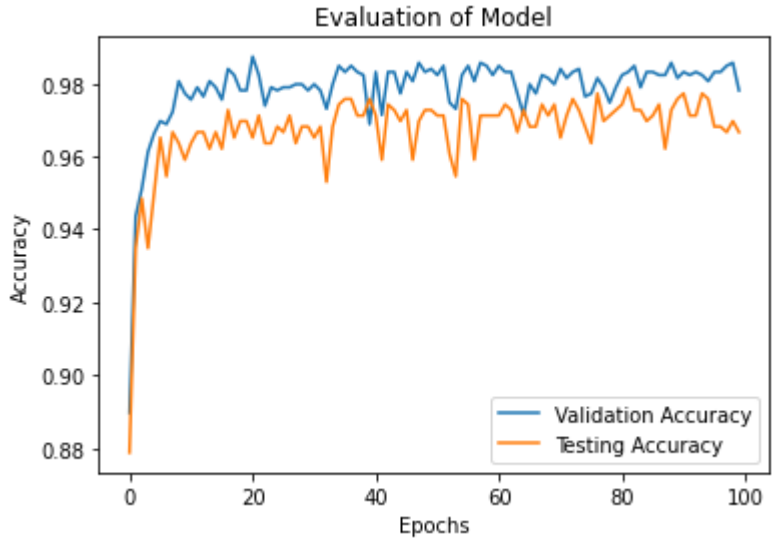


Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

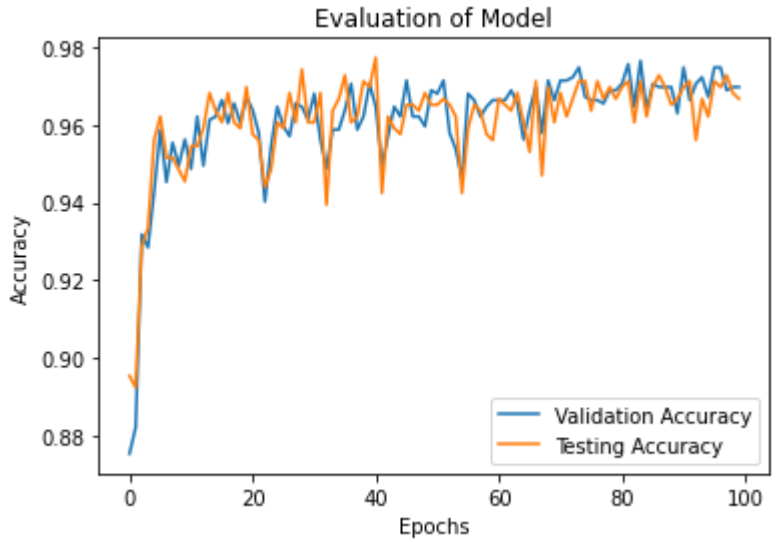
Test average for last 50 epochs: 0.970757575338537
Model saved

INFO:tensorflow:Restoring parameters from ./lenet

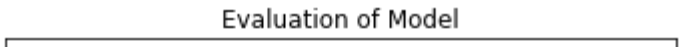
```
INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.967
```

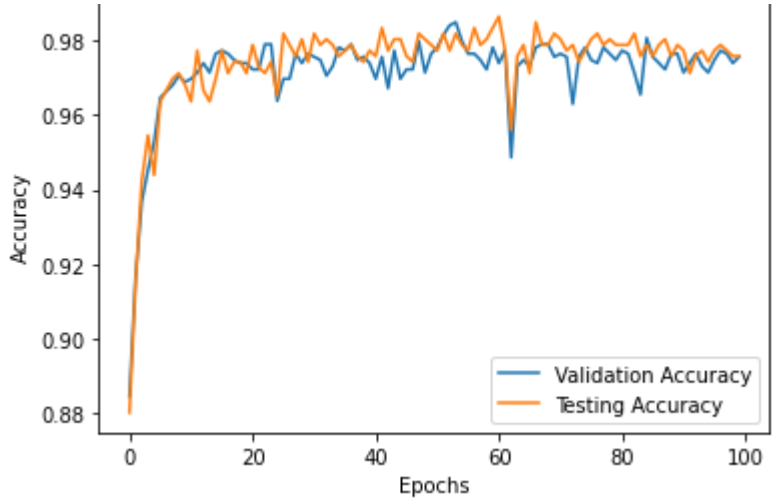



```
Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9651818185791825
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.967
```



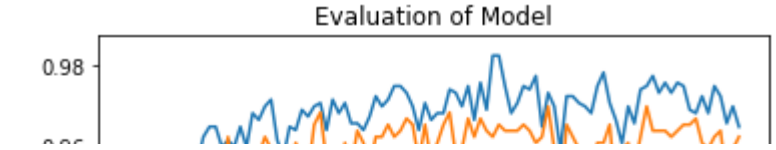

```
Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9780303030664271
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.976
```

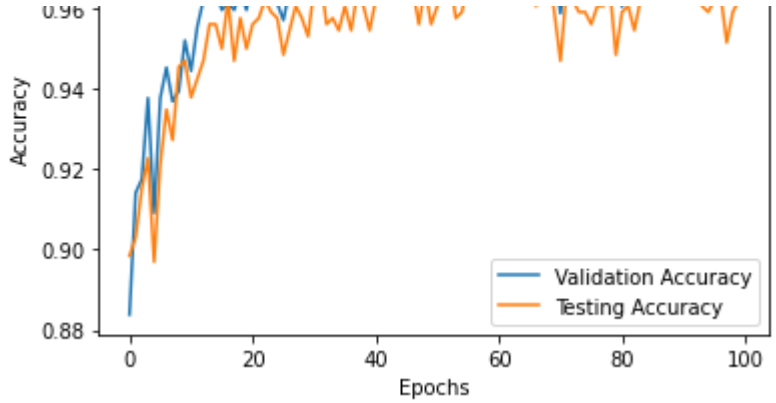




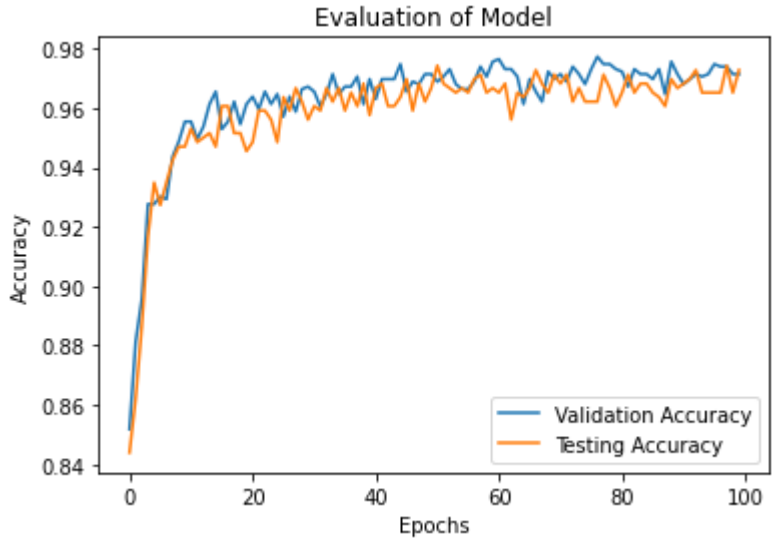

```
Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9793939389026528
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.983
```


```
Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9617878783616155
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.962
```

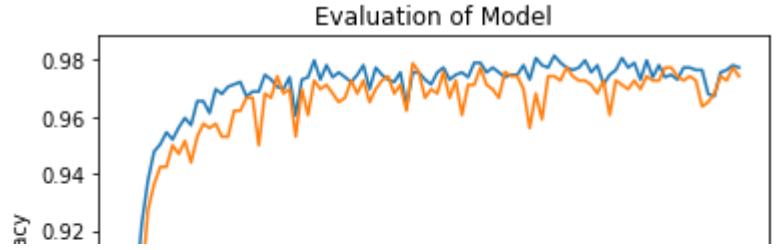


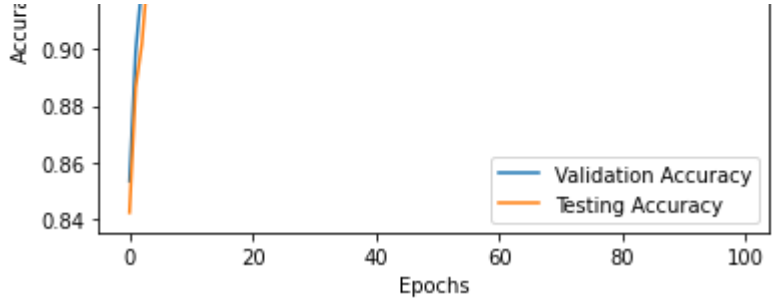



```
Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 256
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9668181821577478
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.973
```

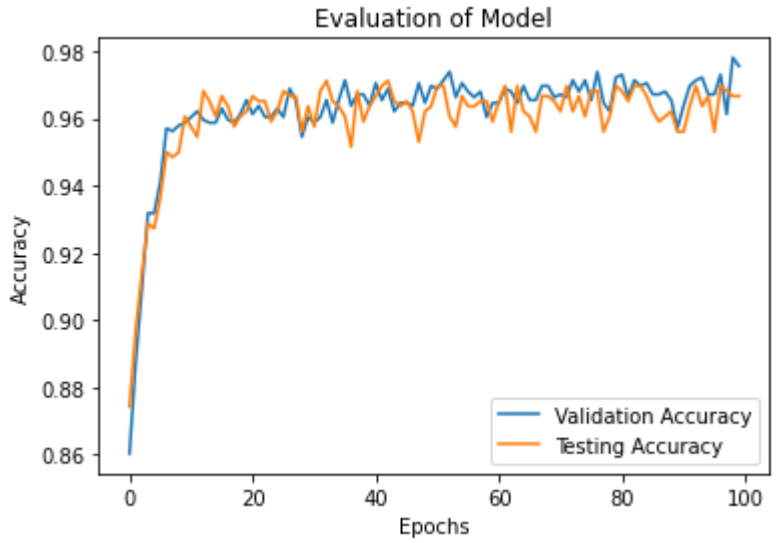



```
Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9711818173148414
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```

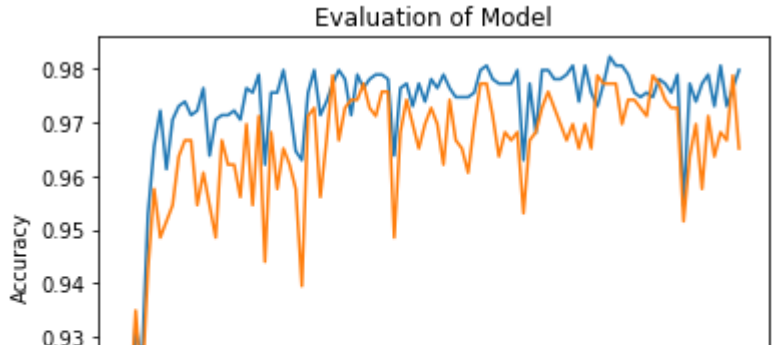


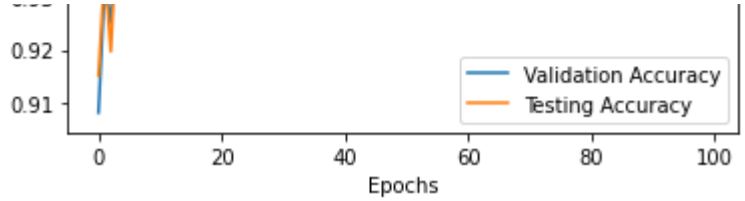



```
Training for:
Learning rate:0.002
Dropout rate: 0.3
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9641818163972913
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.967
```

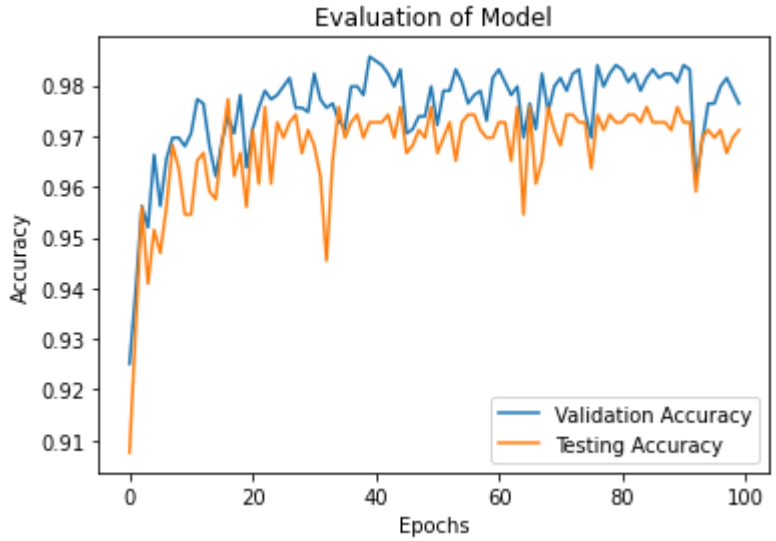



```
Training for:
Learning rate:0.003
Dropout rate: 0.1
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9696060600714249
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.965
```

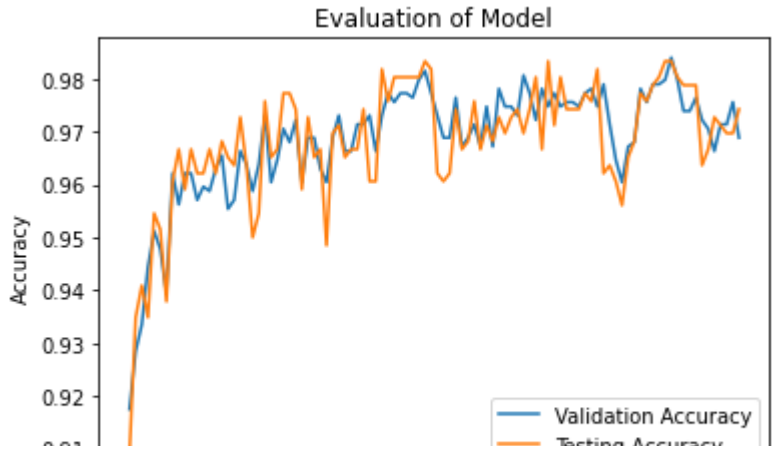


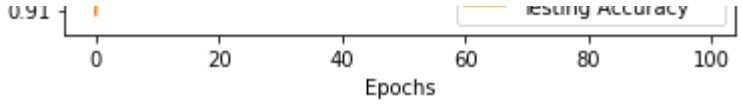



```
Training for:
Learning rate:0.003
Dropout rate: 0.1
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.970848484884609
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```

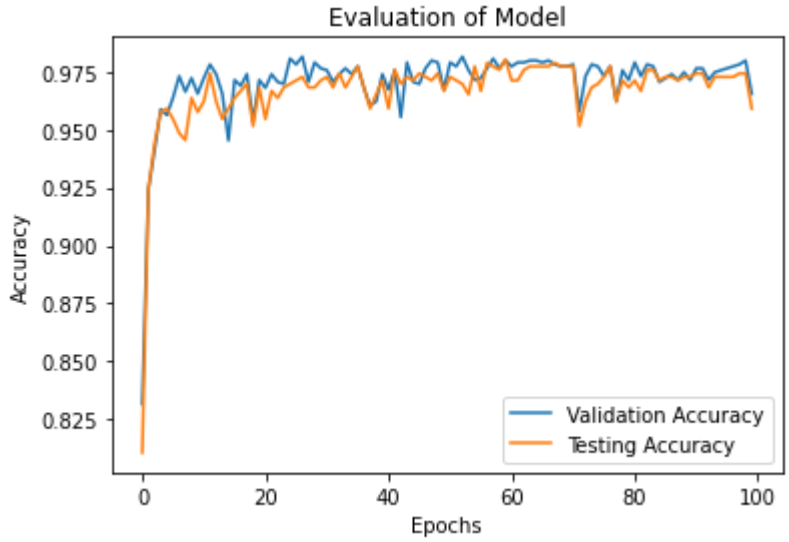



```
Training for:
Learning rate:0.003
Dropout rate: 0.1
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9721515150575927
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```

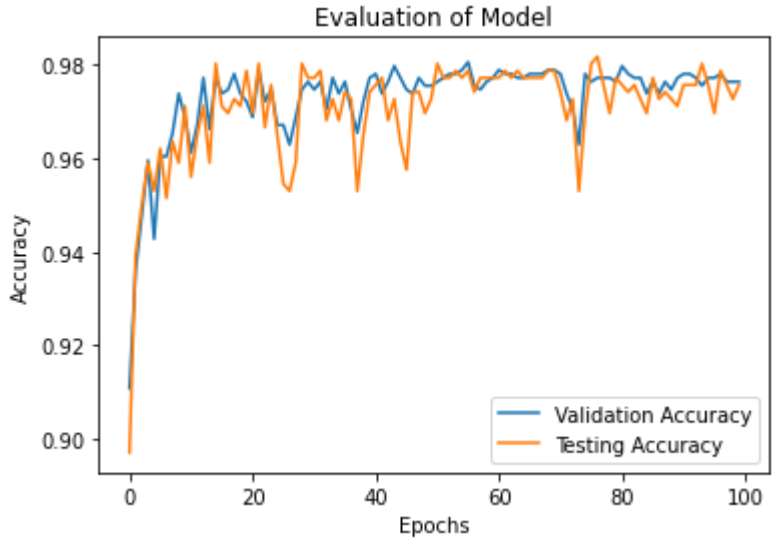




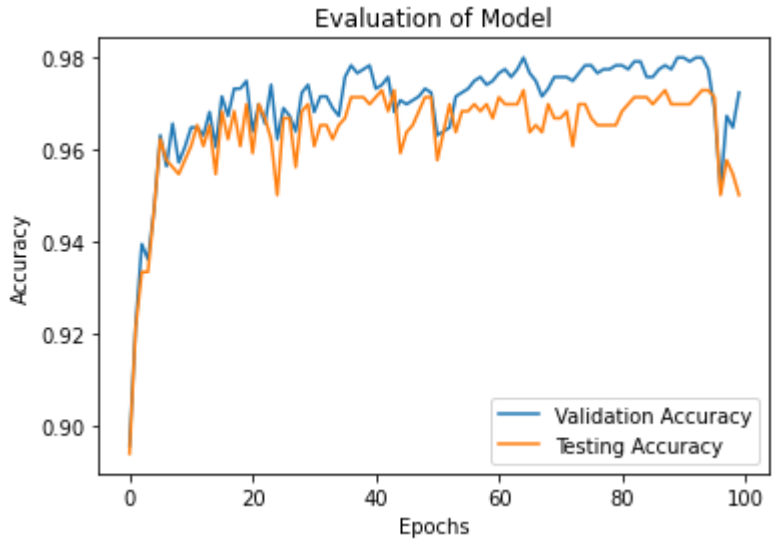

```
Training for:
Learning rate:0.003
Dropout rate: 0.1
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9722727268320142
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.959
```



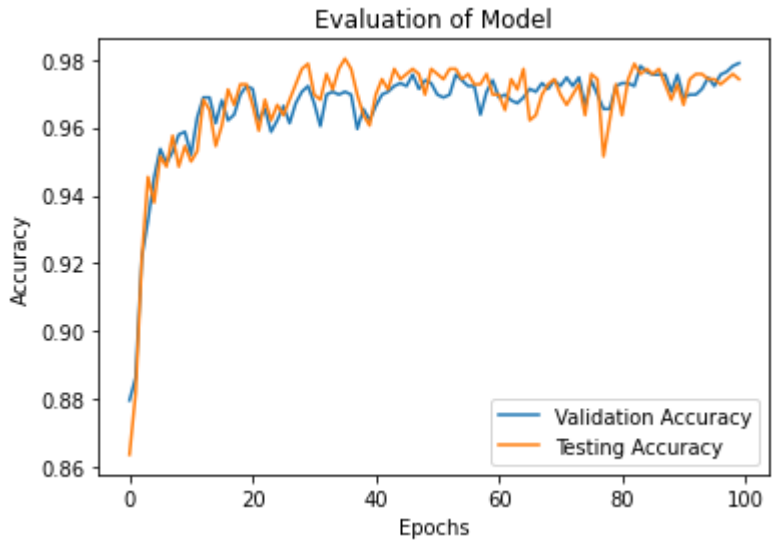

```
Training for:
Learning rate:0.003
Dropout rate: 0.1
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9753939389532263
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.976
```



```
Training for:
Learning rate:0.003
Dropout rate: 0.1
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9670606059811333
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.950
```

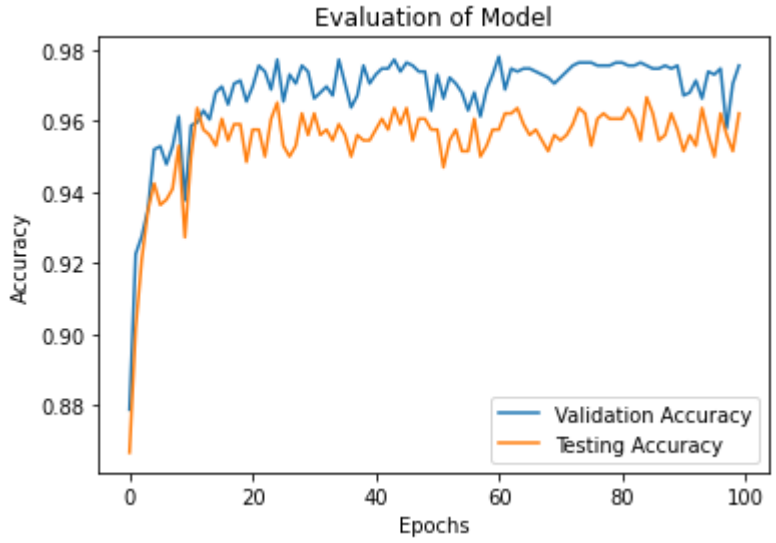



```
Training for:
Learning rate:0.003
Dropout rate: 0.1
Batch size: 256
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9719696978294488
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```

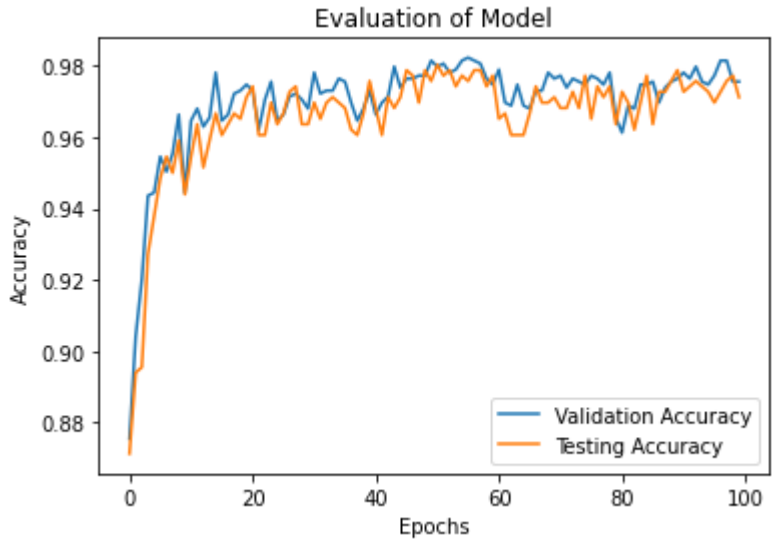



```
Training for:
Learning rate:0.003
Dropout rate: 0.1
```

```
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9574848477912671
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.962
```

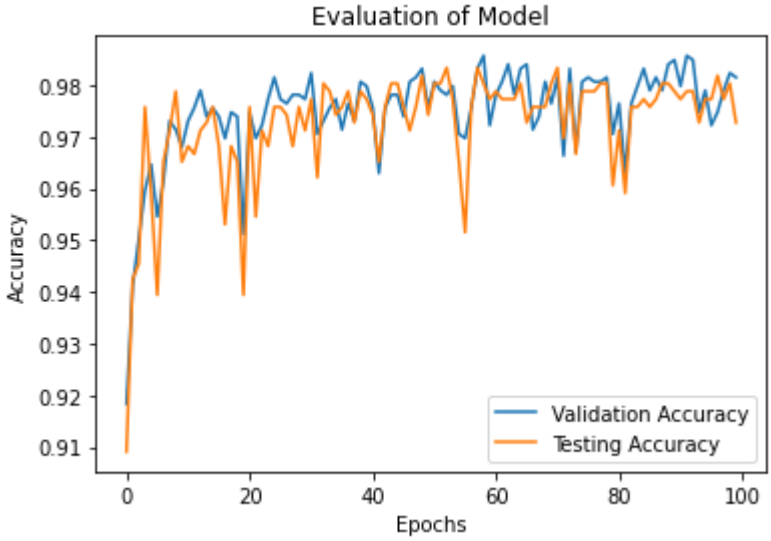



```
Training for:
Learning rate:0.003
Dropout rate: 0.1
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9718484847979113
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```

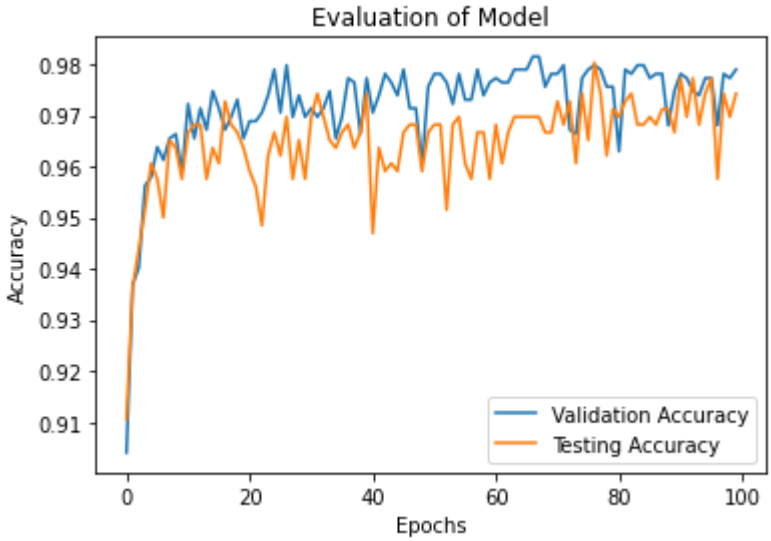



```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>
```

```
Test average for last 50 epochs: 0.9761818179145003
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.973
```

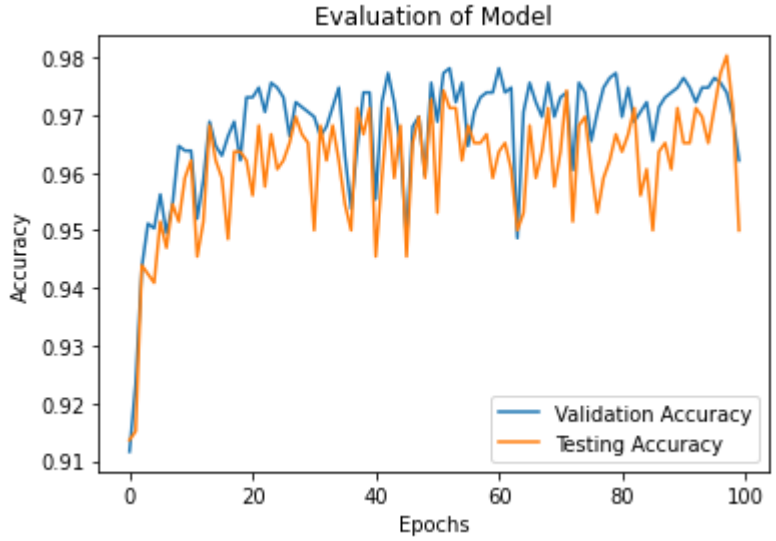



```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9686666660670079
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.974
```

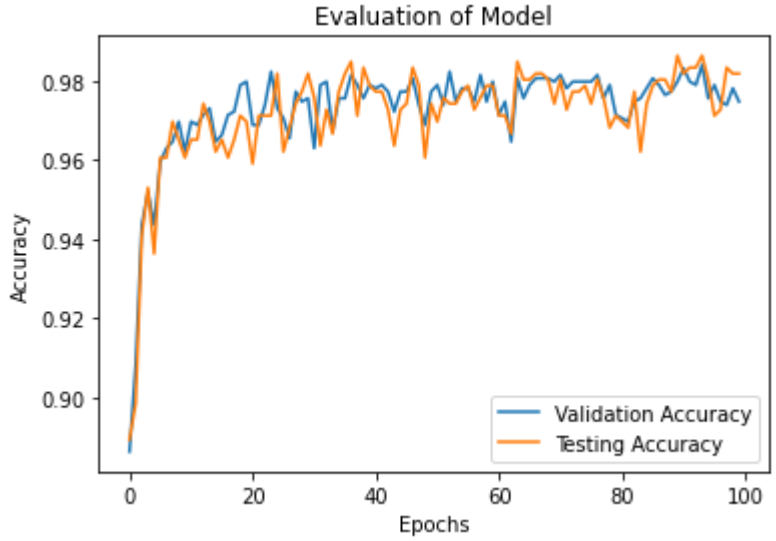



```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.964242424292998
Model saved
```

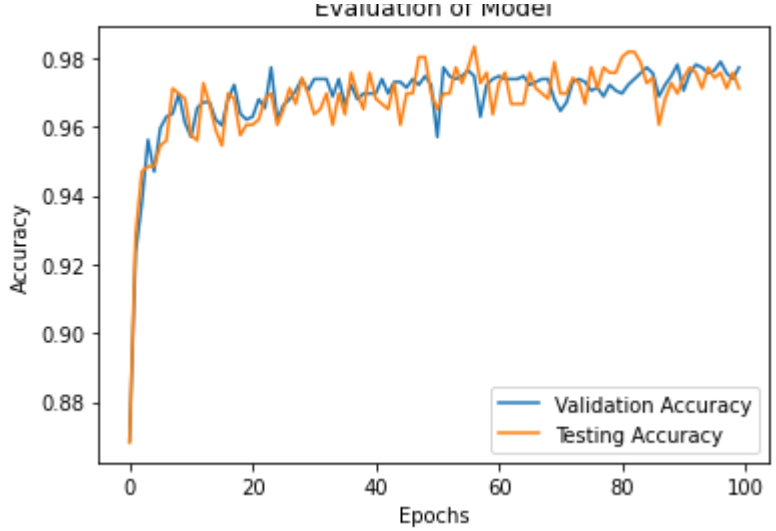
```
INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.950
```



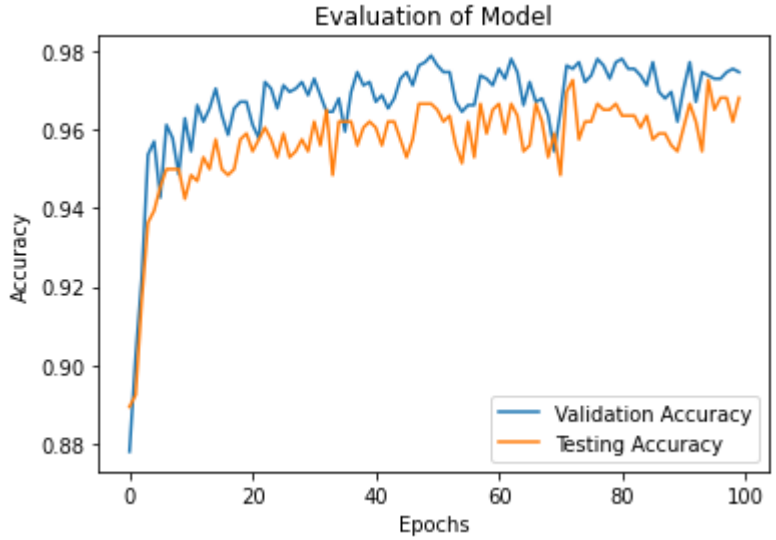

```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9769090903672306
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.982
```



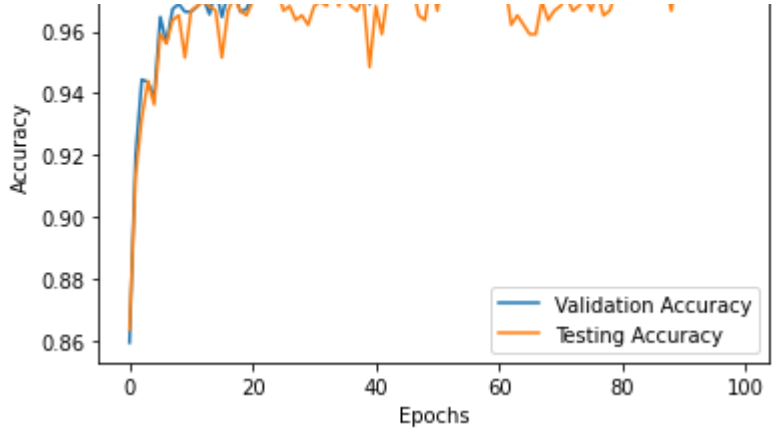

```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9730909088886145
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.971
```



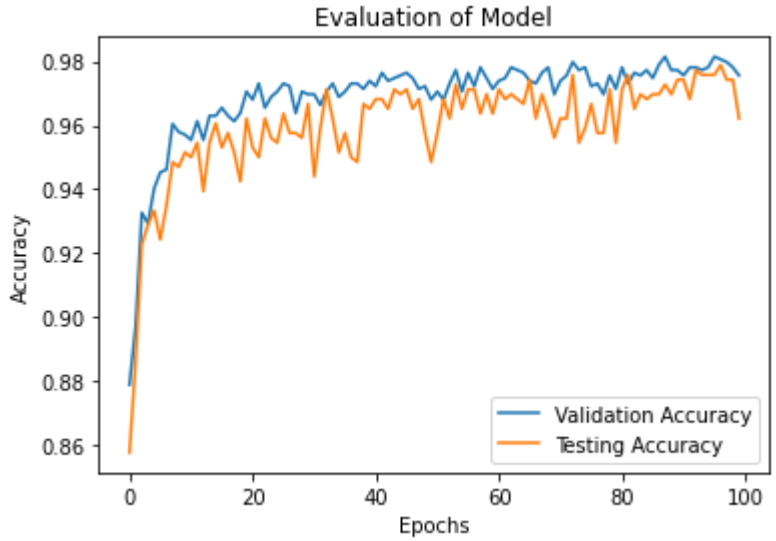

```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9619696971792163
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.968
```



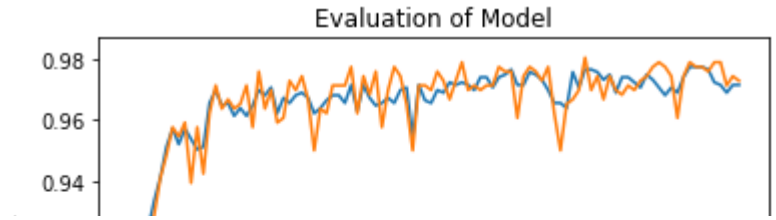

```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 256
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9719393954421534
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.976
```

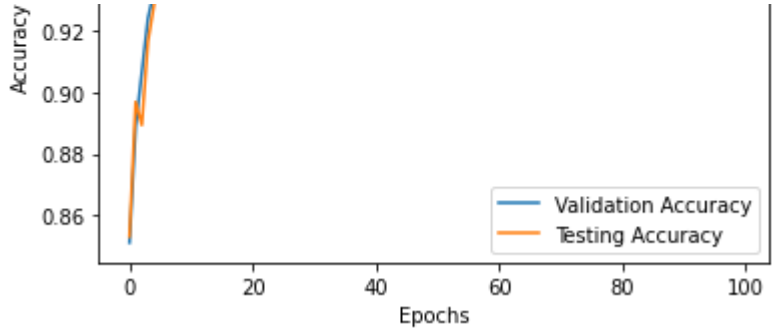



```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9679696972081157
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.962
```

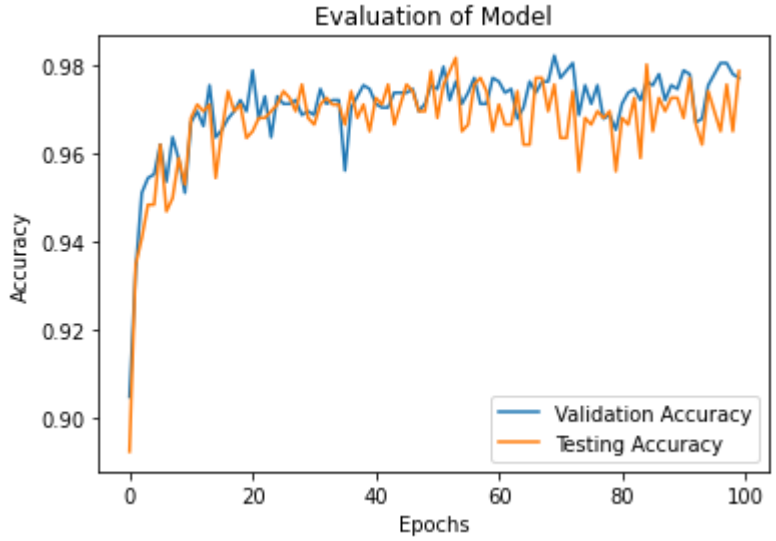



```
Training for:
Learning rate:0.003
Dropout rate: 0.2
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9723636375990781
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.973
```

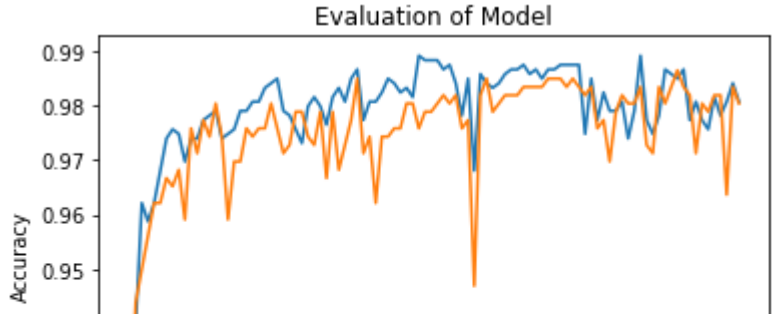


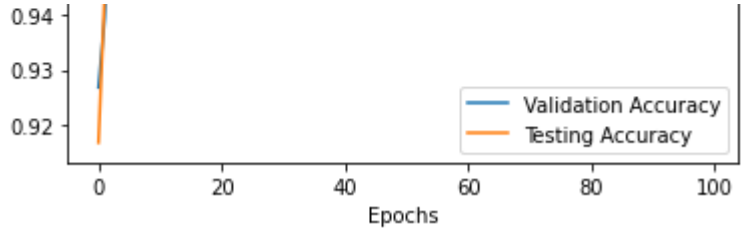



```
Training for:
Learning rate:0.003
Dropout rate: 0.3
Batch size: 64
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9698181814569415
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.979
```

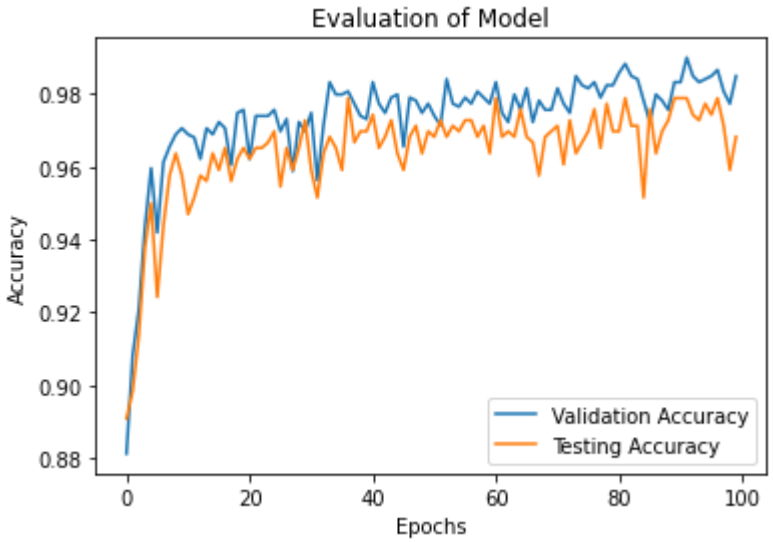



```
Training for:
Learning rate:0.003
Dropout rate: 0.3
Batch size: 64
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9798181813919183
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.980
```

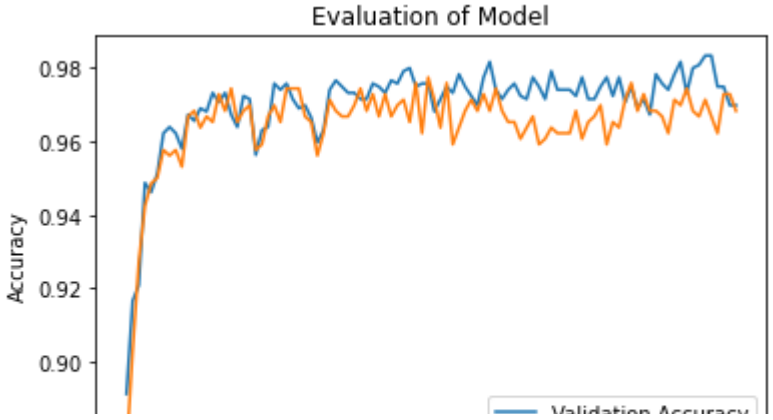


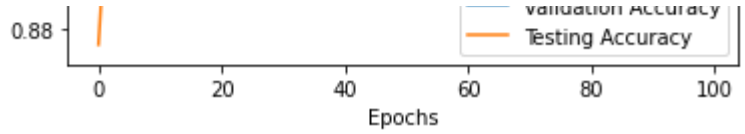



```
Training for:
Learning rate:0.003
Dropout rate: 0.3
Batch size: 64
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9703636360023961
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.968
```

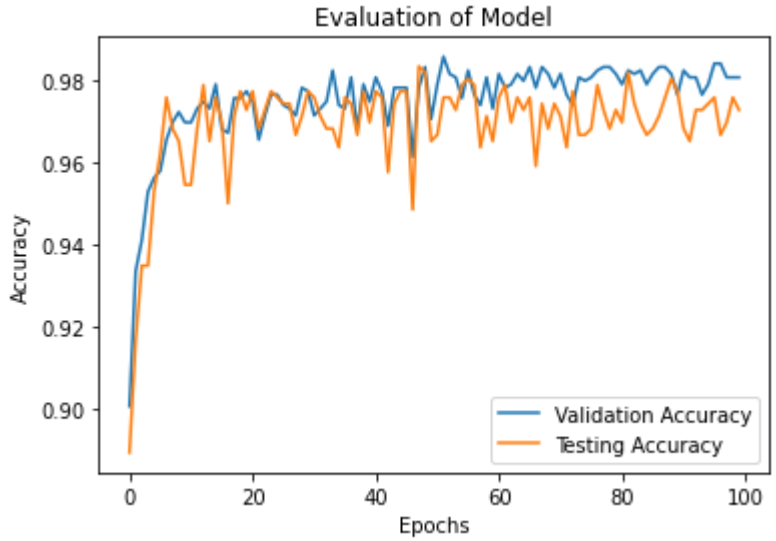



```
Training for:
Learning rate:0.003
Dropout rate: 0.3
Batch size: 128
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9669999995737365
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.968
```

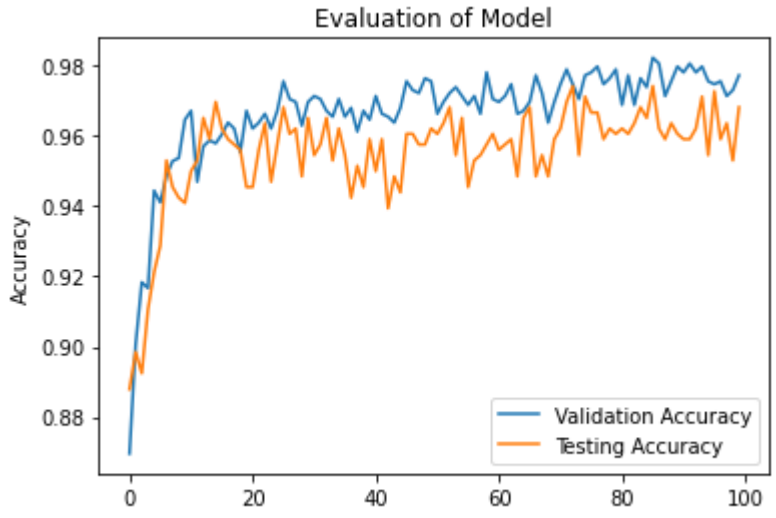


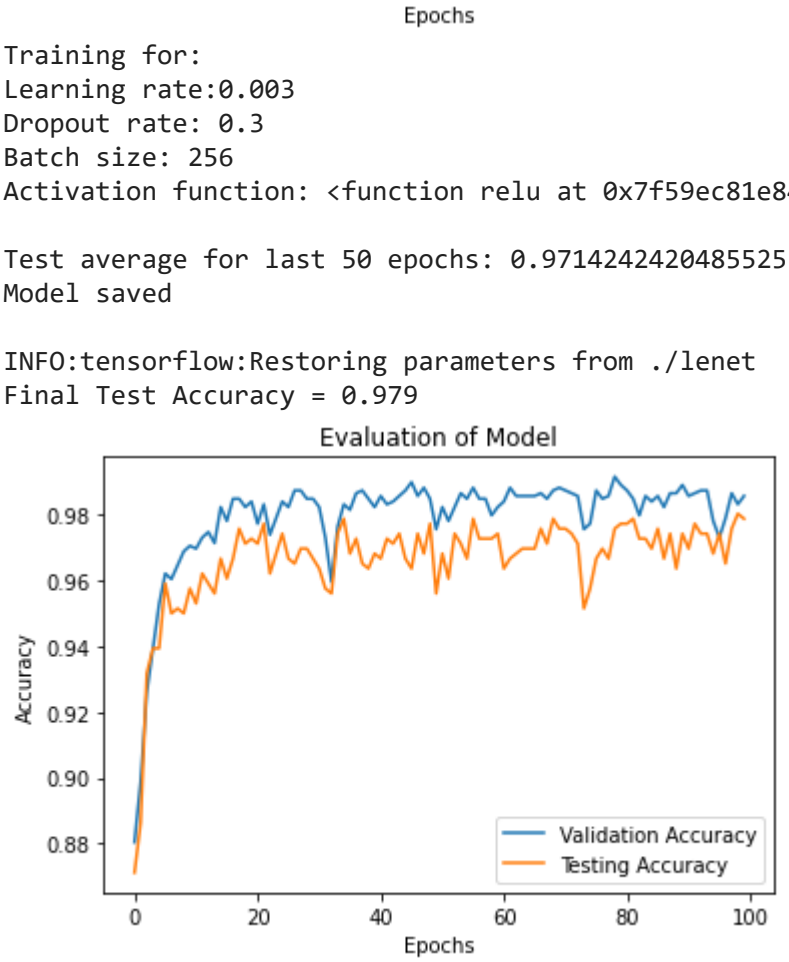



```
Training for:
Learning rate:0.003
Dropout rate: 0.3
Batch size: 128
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9721515146819028
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.973
```




```
Training for:
Learning rate:0.003
Dropout rate: 0.3
Batch size: 128
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.9611515151659649
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.968
```

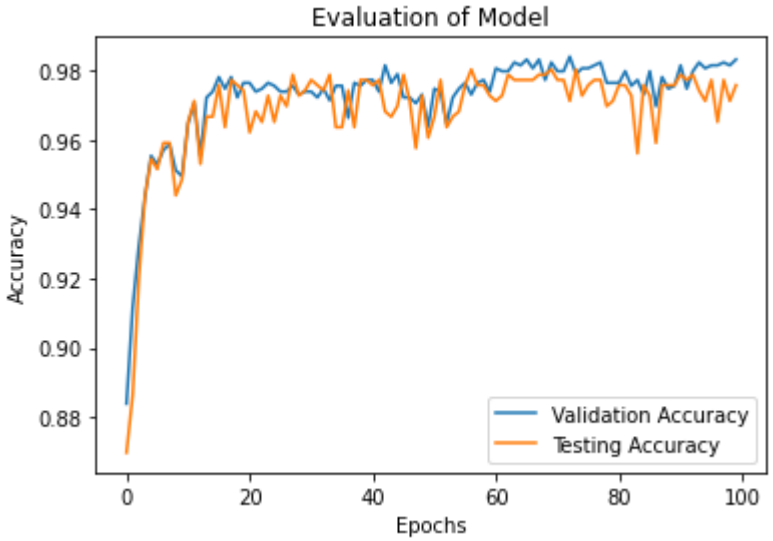





```
Training for:
Learning rate:0.003
Dropout rate: 0.3
Batch size: 256
Activation function: <function relu at 0x7f59ec81e840>

Test average for last 50 epochs: 0.9714242420485525
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.979
```

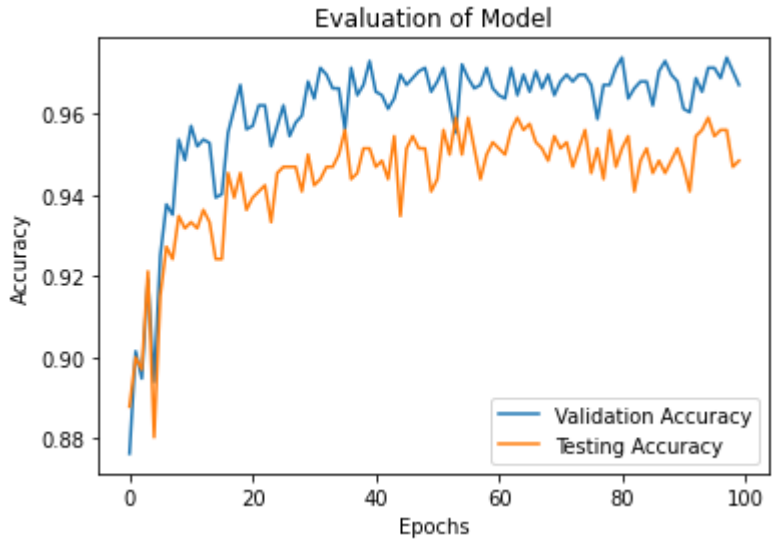


```
Training for:
Learning rate:0.003
Dropout rate: 0.3
Batch size: 256
Activation function: <function relu6 at 0x7f59eb1e7510>

Test average for last 50 epochs: 0.9740909088741648
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.976
```




```
Training for:
Learning rate:0.003
```

```
Dropout rate: 0.3
Batch size: 256
Activation function: <function selu at 0x7f59ec800158>

Test average for last 50 epochs: 0.951272725611022
Model saved

INFO:tensorflow:Restoring parameters from ./lenet
Final Test Accuracy = 0.948
```




```
# Print graph for every run
t = np.arange(0, len(best_testacc), 1)
line1 = plt.plot(t, best_valacc, label='Validation Accuracy')
line2 = plt.plot(t, best_testacc, label='Testing Accuracy')
plt.tick_params(axis='x')
plt.tick_params(axis='y')
plt.legend(loc='lower right')
plt.ylabel("Accuracy")
plt.xlabel("Epochs")
plt.title("Evaluation of Best Model")
plt.show()

print('Best model:')
print('learning rate, dropout rate, batch size, activation function')
print(best_model)
print("Test average for last 50 epochs: " + str(np.mean(best_testacc[50:100])))
```


```
Best model:
learning rate, dropout rate, batch size, activation function
[0.003, 0.3, 64, <function relu6 at 0x7f59eb1e7510>]
Test average for last 50 epochs: 0.9798181813919183
```