ANN:

model = keras.Sequential([

keras.layers.Flatten(input\_shape=(28, 28)),

keras.layers.Dense(128, activation='relu'),

keras.layers.Dense(10, activation='softmax')

])

model.compile(optimizer='adam',

loss='sparse\_categorical\_crossentropy',

metrics=['accuracy'])

CNN:

model = Sequential()

model.add(layers.Conv2D(input\_shape=(256, 256, 3), filters=128,kernel\_size=(7,7), padding="same",activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(128, (5, 5), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (7, 7), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(32, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (5, 5), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(32, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())

model.add(layers.Dense(512, activation='relu'))

model.add(layers.Dense(256, activation='relu'))

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

model.add(layers.Dense(1,activation='sigmoid'))

from tensorflow.keras.optimizers import Adam

model.compile(optimizer='adam',

loss='binary\_crossentropy',

metrics=['accuracy'])

model\_fit = model.fit(train\_gen,

epochs = 20,

verbose = 1,

validation\_data = val\_gen)

RNN:

model = Sequential()

model.add(SimpleRNN(units=32, input\_shape=(1,step), activation="relu"))

model.add(Dense(5, activation="relu"))

model.add(Dense(1))

model.compile(loss='mean\_squared\_error', optimizer='rmsprop',metrics=['accuracy'])

model.summary()

model.fit(trainX,trainY, epochs=100, batch\_size=20, verbose=2,validation\_data = (testX,testY))

trainPredict = model.predict(trainX)

testPredict= model.predict(testX)

predicted=np.concatenate((trainPredict,testPredict),axis=0)

LSTM:

from keras.layers import LSTM

model =Sequential()

model.add(Embedding(input\_dim=len(tokenizer.word\_index)+1, output\_dim=32, input\_length=sequence\_length))

model.add(LSTM(128, input\_shape=(sequences, len(word\_index))))

model.add(Dense(500, activation="relu",kernel\_regularizer=regularizers.l2(0.005)))

model.add(Dense(200, activation="relu",kernel\_regularizer=regularizers.l2(0.005)))

model.add(Dense(units=len(tokenizer.word\_index)+1, activation='softmax',kernel\_regularizer=regularizers.l2(0.005)))

model.compile(optimizer='rmsprop', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])