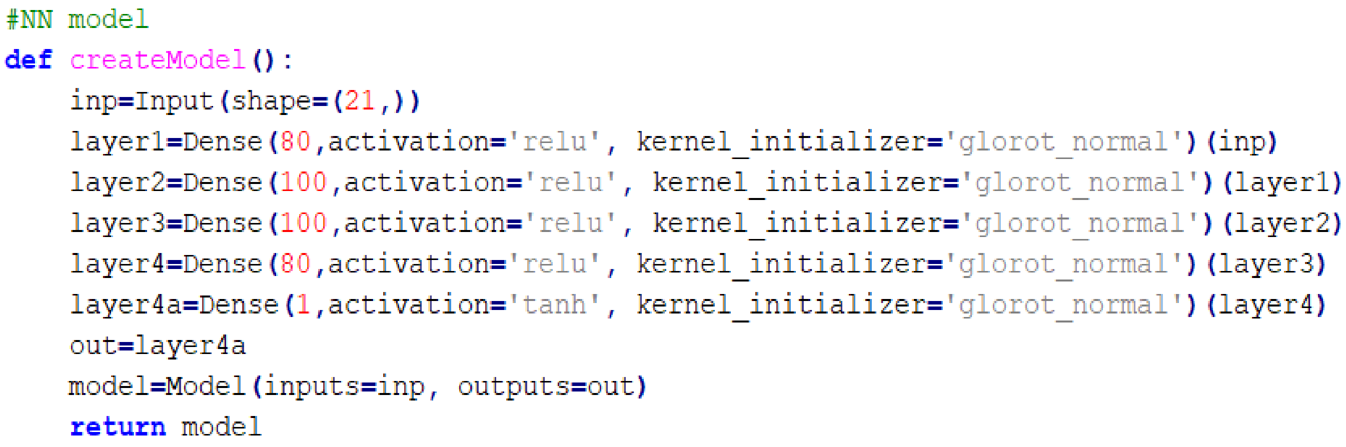
Predicting the system damage of an electrical power grid

Results

Neural Network model

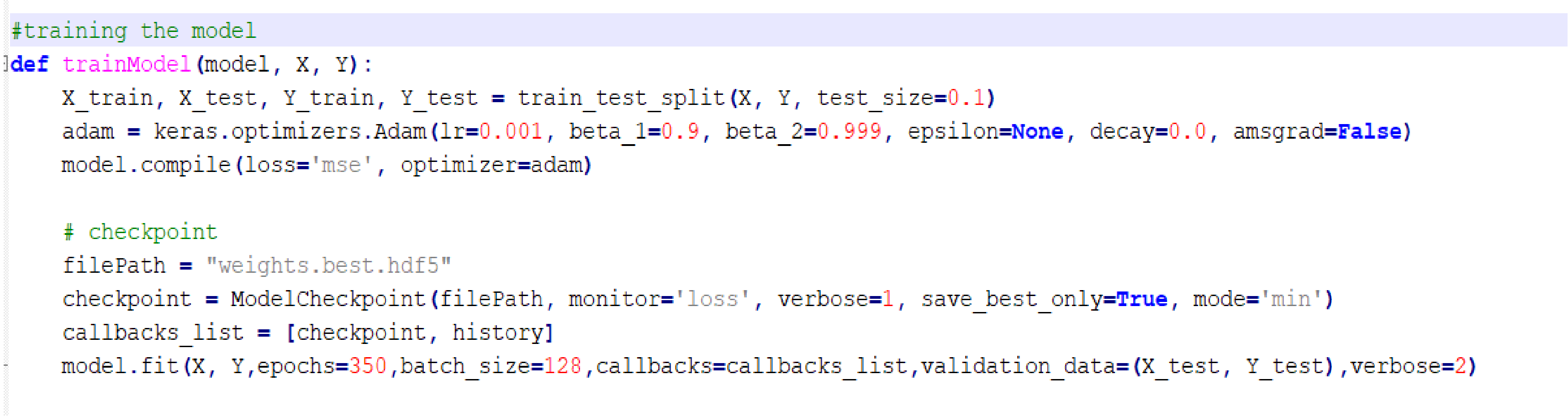
- We use Deep Feed Forward NN for predicting the load loss.
- Model is trained with data for up to (N-6) contingencies, it is then validated and tested with new data for (N-7), (N-8) and (N-9) contingencies
- The predicted output is then compared with the simulator output and MSE is calculated.



```
#NN model
def createModel():
    inp=Input(shape=(21,))
    layer1=Dense(80,activation='relu', kernel_initializer='glorot_normal')(inp)
    layer2=Dense(100,activation='relu', kernel_initializer='glorot_normal')(layer1)
    layer3=Dense(100,activation='relu', kernel_initializer='glorot_normal')(layer2)
    layer4=Dense(80,activation='relu', kernel_initializer='glorot_normal')(layer3)
    layer4a=Dense(1,activation='tanh', kernel_initializer='glorot_normal')(layer4)
    out=layer4a
    model=Model(inputs=inp, outputs=out)
    return model
```

Training

- Epochs: 350
- Batch size: 128
- Optimizer: Adam
- Learning rate: 0.01



```
#training the model
def trainModel(model, X, Y):
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.1)
    adam = keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False)
    model.compile(loss='mse', optimizer=adam)

    # checkpoint
    filePath = "weights.best.hdf5"
    checkpoint = ModelCheckpoint(filePath, monitor='loss', verbose=1, save_best_only=True, mode='min')
    callbacks_list = [checkpoint, history]
    model.fit(X, Y,epochs=350,batch_size=128,callbacks=callbacks_list,validation_data=(X_test, Y_test),verbose=2)
```

ORIGINAL

contingencies	Number of Samples	Prediction time(s)	MSE
N-7	19448	48.68	3.18
N-8	24310	61.87	2.77
N-9	24310	62.39	3.22

Topology1

contingencies	Number of Samples	Prediction time(s)	MSE
N-7	31824	67.07	2.71
N-8	43758	84.04	2.21
N-9	48620	101	2.11

Topology2

contingencies	Number of Samples	Prediction time(s)	MSE
N-7	31824	58.70	2.42
N-8	43758	75.66	2.35
N-9	48620	88.36	2.52