[1]
```
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense, LayerNormalization , Dropout, LSTM, Embedding
from keras.layers.advanced_activations import PReLU, ReLU
from sklearn.model_selection import train_test_split
from tensorflow import keras
import matplotlib.pyplot as plt
#from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import tensorflow as tf
import numpy as np
```

[2]
```
df = pd.read_csv("row_size25_vector_size8000.csv")
```

[3]
```
X = df.iloc[:,7:157]
#X = df[['accelerometer_reading_x_0','accelerometer_reading_y_0','accelerometer_readi
#t = df[['linear_position_x','linear_position_y','linear_position_z','angular_positio
#t = df[['phi_change','theta_change','psi_change']]
t = df[['phi_change']]
```

[4]
```
t.shape
```
```
(8000, 1)
```

[5]
```
in_dim = X.shape[1]
out_dim = t.shape[1]
```

[6]
```
in_dim
```
```
150
```

[7]
```
out_dim
```
```
1
```

[8]
```
X_train, X_test, t_train, t_test = train_test_split(X, t, test_size=0.2)
```

[9]
```
t_train.shape
```
```
(6400, 1)
```

```
[12] model = Sequential()
     model.add(Dense(256,input_dim=in_dim, activation="softplus"))
     model.add(ReLU())
     model.add(Dropout(.32))
     model.add(Dense(128, activation="softplus"))
     model.add(ReLU())
     model.add(LayerNormalization ())
     model.add(Dropout(.25))
     model.add(Dense(64, activation="softplus"))
     model.add(ReLU())
     model.add(LayerNormalization ())
     model.add(Dropout(.1))
     model.add(Dense(32, activation="softplus"))
     model.add(ReLU())
     model.add(LayerNormalization ())
     model.add(Dense(out_dim,activation="softplus"))
     model.compile(loss="mse", optimizer="sgd")
```

```
[13] model.fit(X_train, t_train, epochs=75, batch_size=12, verbose=2)
```

```
Epoch 1/75
534/534 - 4s - loss: 0.0044
Epoch 2/75
534/534 - 1s - loss: 3.3548e-04
Epoch 3/75
534/534 - 1s - loss: 2.5875e-04
Epoch 4/75
534/534 - 1s - loss: 2.9737e-04
Epoch 5/75
534/534 - 1s - loss: 1.9211e-04
Epoch 6/75
534/534 - 1s - loss: 1.2780e-04
Epoch 7/75
534/534 - 1s - loss: 9.2511e-05
Epoch 8/75
534/534 - 1s - loss: 3.0580e-04
Epoch 9/75
534/534 - 1s - loss: 5.5662e-05
Epoch 10/75
534/534 - 1s - loss: 4.8231e-05
Epoch 11/75
534/534 - 1s - loss: 5.4477e-05
Epoch 12/75
534/534 - 1s - loss: 4.6763e-05
Epoch 13/75
534/534 - 1s - loss: 3.9239e-05
Epoch 14/75
534/534 - 1s - loss: 4.1927e-05
Epoch 15/75
534/534 - 1s - loss: 4.0162e-05
Epoch 16/75
534/534 - 1s - loss: 2.9078e-05
Epoch 17/75
534/534 - 1s - loss: 7.1709e-05
Epoch 18/75
534/534 - 1s - loss: 3.3173e-05
Epoch 19/75
```

```
534/534 - 1s - loss: 2.6474e-05
Epoch 20/75
534/534 - 1s - loss: 3.9822e-05
Epoch 21/75
534/534 - 1s - loss: 4.0852e-05
Epoch 22/75
534/534 - 1s - loss: 3.5169e-05
Epoch 23/75
534/534 - 1s - loss: 1.9355e-05
Epoch 24/75
534/534 - 1s - loss: 2.3321e-05
Epoch 25/75
534/534 - 1s - loss: 1.0175e-04
Epoch 26/75
534/534 - 1s - loss: 2.4923e-05
Epoch 27/75
534/534 - 1s - loss: 2.4962e-05
Epoch 28/75
534/534 - 1s - loss: 2.5857e-05
Epoch 29/75
534/534 - 1s - loss: 2.4550e-05
Epoch 30/75
```

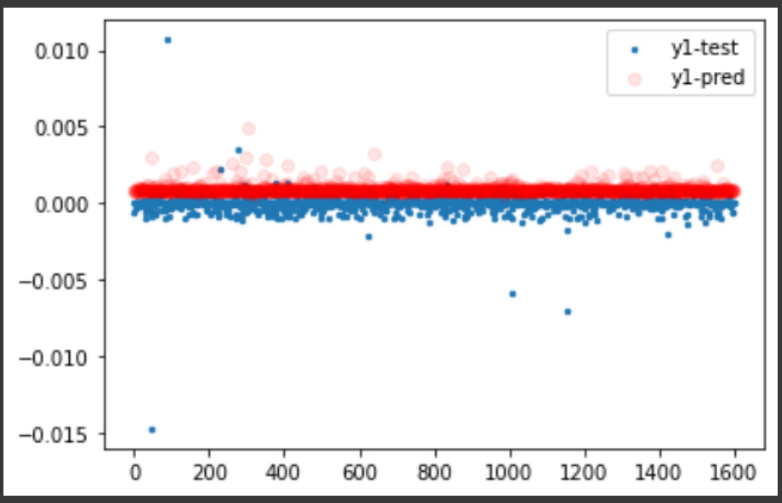
[14]
```
ypred = model.predict(X_test)
print("y1 MSE: ", mean_squared_error(t_test.iloc[:, 0], ypred[:,0]))
#print("y2 MSE: ", mean_squared_error(t_test.iloc[:, 1], ypred[:,1]))
#print("y3 MSE: ", mean_squared_error(t_test.iloc[:, 2], ypred[:,2]))
```

```
y1 MSE:  1.3592664811770315e-06
```

[15]
```
x_ax = range(len(X_test))

plt.scatter(x_ax, t_test.iloc[:, 0],  s=6, label="y1-test")
plt.scatter(x_ax, ypred[:,0], label="y1-pred",c="red",alpha = 0.1)

plt.legend()
plt.show()
```

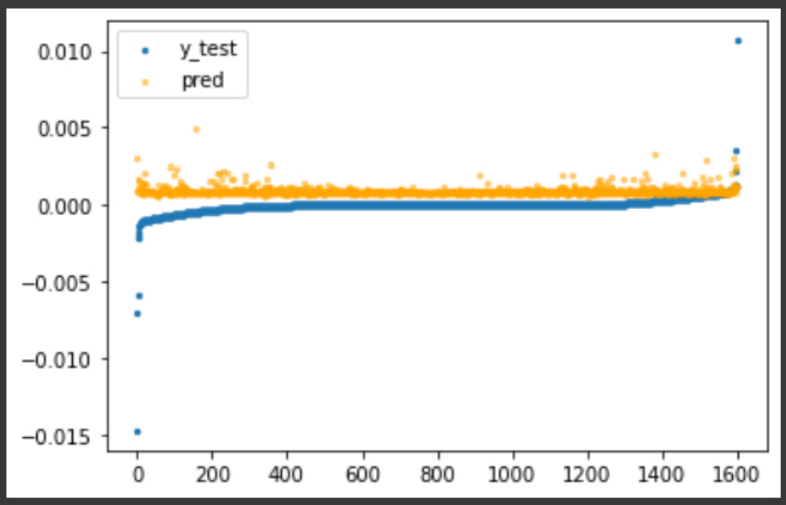



```
x_ax = range(len(X_test))
```

```
y_test_index = np.argsort(t_test.iloc[:, 0], axis=0).to_numpy()

f = plt.figure()
plt.scatter(x_ax, t_test.iloc[y_test_index],  s=6, label="y_test")
plt.scatter(x_ax, ypred[y_test_index], s=6, label="pred",c="orange", alpha=0.5)
#plt.ylim(t_test.iloc[y_test_index[0]].to_numpy()[0])
plt.legend()
plt.show()

f.savefig("foo.pdf", bbox_inches='tight')
```




```
[ ]
```

✓ 0s completed at 8:20 PM