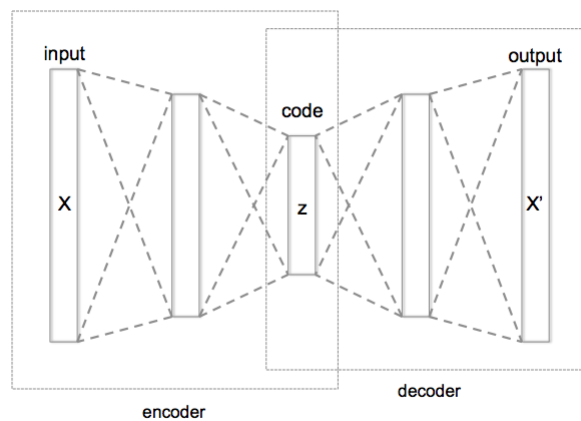


Introduction to 2 Dimensional LSTM Autoencoder



Adnan Karol [Follow](#)
Oct 17, 2020 · 3 min read

This is a continuation of my earlier repository where I use 1D LSTM for Autoencoder and Anomaly Detection which can be found [here](#).



AUTOENCODERS Structure

Here I extend the topic to LSTM Autoencoder for 2D Data. Creating a separate post about this as LSTM tends to become really tricky when speaking of inputs.

An amazing blog explaining the INPUTS and OUTPUTS can be found by [@Timur Bikmukhametov here](#).

Let's see some codes for understanding 2D LSTM Autoencoders.

We are going to use simple data that I created for test for this purpose and the same can be found with the Source code at GitHub [here](#).

Let's get started !!

Import Dependencies

```
import pandas as pd
import numpy as np

from tensorflow import keras
from tensorflow.python.keras.layers import Input, Dense, RepeatVector,
TimeDistributed, Dense, Dropout, LSTM
from tensorflow.python.keras.models import Sequential

import matplotlib.pyplot as plt
%matplotlib inline

import sklearn
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

Read The Data

```
df = pd.read_csv('../Data/Sample.csv', parse_dates=['Time Stamp'],
index_col='Time Stamp')
df.head(n=5)
```

	Class	Confidence
Time Stamp		
1602438412	1	0.84
1602438413	2	0.73
1602438414	3	0.51
1602438415	4	0.49
1602438416	5	0.97

Dataset Used for Test

Create Data for LSTM Input (Series of Data i.e here create a series of data with time_steps number of elements)

```
def create_dataset(X, y, time_steps=1):
    Xs, ys = [], []
    for i in range(len(X) - time_steps):
        v = X.iloc[i:(i + time_steps)].values
        Xs.append(v)
        ys.append(y.iloc[i + time_steps])
    return np.array(Xs), np.array(ys)

# Timesteps will define how many Elements we have
TIME_STEPS = 5

X_train, y_train = create_dataset(df, df, TIME_STEPS)

print(X_train.shape)
```

The shape of X_train is (20, 5, 2), with each block of input having 5 samples with 2 features each.

If you understand this step then the rest is much easier.

#Size of Input Data is n_samples * timesteps * n_features

Create the LSTM AUTOENCODER MODEL

```
model = Sequential()
model.add(LSTM(128, input_shape=(X_train.shape[1],
X_train.shape[2])))
model.add(Dropout(rate=0.2))
model.add(RepeatVector(X_train.shape[1]))
model.add(LSTM(128, return_sequences=True))
model.add(Dropout(rate=0.2))
model.add(TimeDistributed(Dense(X_train.shape[2])))
model.compile(optimizer='adam', loss='mae')
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 128)	67072
dropout (Dropout)	(None, 128)	0
repeat_vector (RepeatVector)	(None, 5, 128)	0
lstm_1 (LSTM)	(None, 5, 128)	131584
dropout_1 (Dropout)	(None, 5, 128)	0
time_distributed (TimeDistri	(None, 5, 2)	258
Total params: 198,914		
Trainable params: 198,914		
Non-trainable params: 0		

Summary of Model

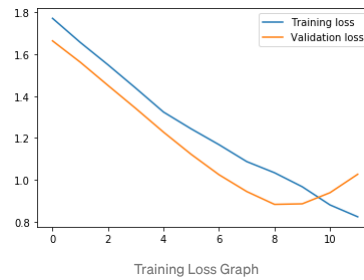
Train the Model

```
history = model.fit(X_train, X_train, epochs=200, batch_size=32,
validation_split=0.1)
```

Visualize Training

```
plt.plot(history.history['loss'], label='Training loss')
plt.plot(history.history['val_loss'], label='Validation loss')
```

```
plt.legend();
```



Test the Model

Now, Comes the Most important phase of testing how close we are to recreating the input with our model which is hardly trained with data.

```
#Create a new test data close to what model has seen and check the MSE
test1 = np.array([[ 1, 0.7],
                  [2, 0.86],
                  [2.9, 0.85],
                  [4.1, 0.64],
                  [5, 0.89]])

test = np.reshape(test1, newshape=(-1, 5, 2))
y = model.predict(test)
y = y.reshape(5, 2)
print(" THE MSE IS : ", sklearn.metrics.mean_squared_error(test1, y))
print("The Recreated Output is : ", y)
```

Let's check the MSE and what our Model has recreated.

```
THE MSE IS : 0.09770384336959057
The Recreated Output is : [[1.1208125 0.33779058]
 [2.170816 0.5113479 ]
 [3.0839798 0.5797549 ]
 [3.819326 0.5880815 ]
 [4.376938 0.5675875 ]]
```

Here the model is able to recreate the input with low Error

Now let us check with test data with which the model is not trained.

```
#Create a new test data close to what model has NOT seen and check the MSE
test2 = np.array([[ 1, 0.74],
                  [6, 0.60],
                  [7, 0.96],
                  [8, 0.42],
                  [5, 0.85]])

test = np.reshape(test2, newshape=(-1, 5, 2))
y = model.predict(test)
y = y.reshape(5, 2)
print(" THE MSE IS : ", sklearn.metrics.mean_squared_error(test2, y))
print("The Recreated Output is : ", y)
```

Let's check the MSE and what our Model has recreated.

```
THE MSE IS : 2.5207483415118306
The Recreated Output is : [[1.6618627 0.4883529 ]
 [3.1379414 0.71685326]
 [4.2798247 0.7910608 ]
 [5.0690384 0.79329824]
 [5.580368 0.7696756 ]]
```

Here the model is NOT able to recreate the Input as it's not trained on such data and cant extract its features.

So we have learned how to make a 2 Dimensional LSTM Autoencoder. The applications are many which I have listed in my previous blogs like :

- 1. Removal of Noise
- 2. Feature extraction (Use only Encoder part)
- 3. Anomaly Detection


Github Repository: <https://github.com/adnanmushtaq1996/2D-LSTM-AUTOENCODER>

Hope you Enjoyed Reading it :)

Sign up for Analytics Vidhya News Bytes

By Analytics Vidhya

Latest news from Analytics Vidhya on our Hackathons and some of our best articles! [Take a look.](#)

 [Get this newsletter](#)

Emails will be sent to varanasi.sreekanth65@gmail.com. [Not you?](#)

2d

Lstm

Autoencoder

Time Series Data

Python

Learn more.

Medium is an open platform where 170 million readers come to find insightful and dynamic thinking. Here, expert and undiscovered voices alike dive into the heart of any topic and bring new ideas to the surface. [Learn more](#)

Make Medium yours.

Follow the writers, publications, and topics that matter to you, and you'll see them on your homepage and in your inbox. [Explore](#)

Write a story on Medium.

If you have a story to tell, knowledge to share, or a perspective to offer — welcome home. It's easy and free to post your thinking on any topic. [Start a blog](#)

