MohammadFneish7 / **Keras_LSTM_Diagram**

Understanding Keras Recurrent Nets' structure and data flow (mainly LSTM) in a single diagram.

Keras_LSTM_Diagram

Understanding Keras Recurrent Nets' structure and data flow (mainly LSTM) in a single diagram.

Actually as I was working on understanding how `Recurrent Neural Networks` really work and what gives these special network architectures this high power and efficiency, especially when working with sequence datasets, I found many difficulties to get the whole concept clearly into my mind. I was very familiar with `Convolutional Neural Networks` and fully-connected `Feed-Forward` architectures, where I had been working with these networks for a long time ago, and I can simply describe it as an easy-to-cut cake. However, when talking about `Recurrent Nets` *(e.g. LSTM's, GRU's ...)* things gets much more complex.

So, when it came into building up my own `LSTM` model for the first time, I decided to use **Keras framework** *(because I was very familiar with)*, and I realized then that the worst and the hardest part was about understanding how to prepare and transform my input data to match the Keras model input expectations, and how to transform my training and validation labels to match the output of the network for validation and testing. Besides, when working with LSTM's you will find that there are a lot of network-special parameters that you must understand and take care of in order to start the game.

Finally after having a clear and sharp view of this problem, and because all of what I passed through, *I decided to summarize the main flow of Keras LSTM networks in the this diagram for the public benefit*.

Please note that the following diagram describes Keras LSTM layers, however the same diagram, as it is, is applicable for GRU's, with a minor difference between both structures in the inner shape of the processing units *(nodes)*.

For Keras LSTM code example, I think that Mr.Jason Brownlee has a great blog here that worth to check.

Diagram

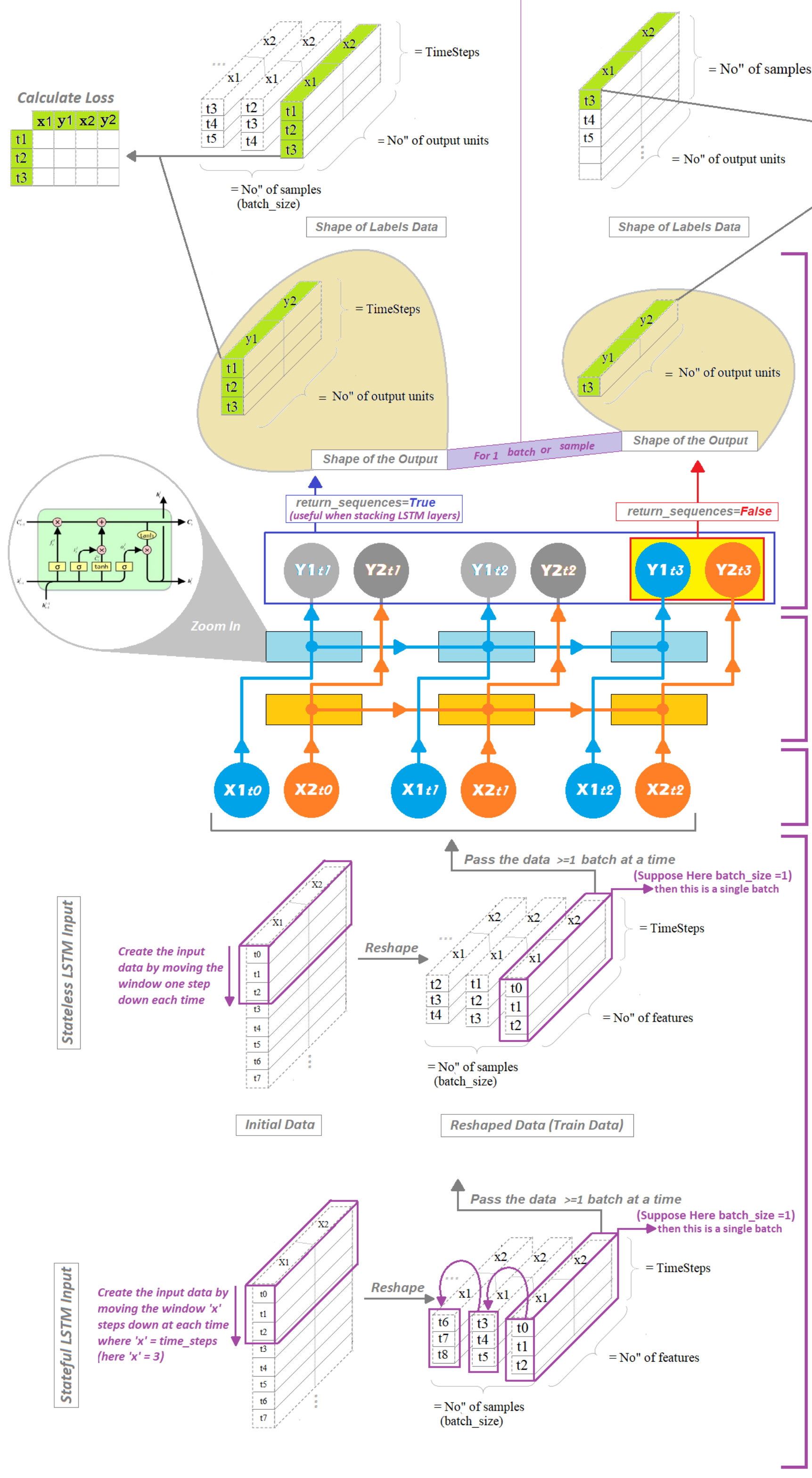
Please try to read the diagram from bottom to top for better flow coherence.

README.md

Understanding Keras LSTM Input & Output

Here we are trying to predict the next batch of sequence of some 'y' variables based on the previous batch of sequence of some features

Here we are trying to predict the next value in a sequence of some 'y' variables based on the previous batch of sequence of some features



Calculate Loss
x1 y1 x2 y2
t1
t2
t3
= TimeSteps
= No" of output units
= No" of samples (batch_size)
Shape of Labels Data
= No" of samples
= No" of output units
Shape of Labels Data
Calculate Loss
x1 y1 x2 y2
t3
= TimeSteps
= No" of output units
Shape of the Output
For 1 batch or sample
= No" of output units
Shape of the Output
return_sequences=True (useful when stacking LSTM layers)
return_sequences=False
Zoom In
Y1 t1
Y2 t1
Y1 t2
Y2 t2
Y1 t3
Y2 t3
X1 t0
X2 t0
X1 t1
X2 t1
X1 t2
X2 t2
Pass the data >=1 batch at a time
(Suppose Here batch_size =1) then this is a single batch
Stateless LSTM Input
Create the input data by moving the window one step down each time
Reshape
= TimeSteps
= No" of features
= No" of samples (batch_size)
Initial Data
Reshaped Data (Train Data)
Pass the data >=1 batch at a time
(Suppose Here batch_size =1) then this is a single batch
Stateful LSTM Input
Create the input data by moving the window 'x' steps down at each time where 'x' = time_steps (here 'x' = 3)
Reshape
= TimeSteps
= No" of features
= No" of samples (batch_size)

Output Layer
Holds a number of output nodes that equals:

- **if(return_sequences = True)**
  output units (No" of Y variables we are predicting) * TimeSteps
  *(Here we have 6 Nodes = 2 output units * 3 steps)*
- **if(return_sequences = False)**
  output units (No" of Y variables we are predicting)
  (Here we have 2 Nodes = 2 output units)

Note that the number of output units could be passed as a parameter into the LSTM layer and could be any number that is equal to the number of variables we are predicting,thus it's not related to the number of input features.

Hidden Layers
Represents a number of RNN processing layers such that the number of these layers equals to the number of features and the number of nodes in each layer equals to the TimeSteps
(Here we have 2 layers for X1 & X2, and 3 nodes in each)

Input Layer
This is obviously the layer that accepts the input where it always requires 3D input as: ***(No" of batches or samples, TimeSteps, No" of features)***

Input Data
This is how the train data should be reshaped to look like a 3 dimensional array as an input for the Keras LSTM layer
(Note that the shape of the labels data should be identical to the shape of the output in the output layer)

Note: stateless LSTM's are used when each batch of your data is not related to the data in the next batch, where each batch represents a standalone inner related sequence.

While statful LSTM's are used when data batches are highly related, and the whole data is a one hot sequence continuous through time. In this case keras keeps the state of the network through batches.

In case of stateful LSTM's it is necessary to turn off 'suffle' when fitting the model, provide the 'batch_size' for the input layer, and manually reset the model state after each epoch (where the last batch is definitely not related to the first one).

Note: the above output example belongs to stateless LSTM input where the shape of the ouput in case of stateful LSTM input would be a bit different.

General Notes:

1. *To build a LSTM layer (as the first layer in the model) use as an example:*
   model = keras.Sequential()
   model.add(layers.LSTM(units=LSTM_output_units, batch_input_shape=(batch_size, time_steps, number_of_features), stateful=True, return_sequences=False))

2. *The above model is of single layer (LSTM) with no Dense layer at the end.*

3. *In the above example we takes the next value (or the next sequence in case of return_sequences=True) of 'x1' & 'x2', as our label (the value we are trying to let the LSTM predict), however in some problems it could be the next value (or the next sequence in case of r eturn_sequences=True) of another variable from outside the input features.*

License

Releases

No releases published

Packages

No packages published