Open in Colab

T81-558: Applications of Deep Neural Networks

Module 10: Time Series in Keras

- Instructor: Jeff Heaton, McKelvey School of Engineering, Washington University in St. Louis
- For more information visit the class website.

Module 10 Material

- Part 10.1: Time Series Data Encoding for Deep Learning [Video] [Notebook]
- **Part 10.2: Programming LSTM with Keras and TensorFlow** [Video] [Notebook]
- Part 10.3: Text Generation with Keras and TensorFlow [Video] [Notebook]
- Part 10.4: Introduction to Transformers [Video] [Notebook]
- Part 10.5: Transformers for Timeseries [Video] [Notebook]

Google CoLab Instructions

The following code ensures that Google CoLab is running the correct version of TensorFlow. Running the following code will map your GDrive to `/content/drive`.

In [1]:
```
try:
    from google.colab import drive
    drive.mount('/content/drive', force_remount=True)
    COLAB = True
    print("Note: using Google CoLab")
    %tensorflow_version 2.x
except:
    print("Note: not using Google CoLab")
    COLAB = False
```

```
Mounted at /content/drive
Note: using Google CoLab
```

Part 10.2: Programming LSTM with Keras and TensorFlow

So far, the neural networks that we've examined have always had forward connections. Neural networks of this type always begin with an input layer connected to the first hidden layer. Each hidden layer always connects to the next hidden layer. The final hidden layer always connects to the output layer. This manner of connection is why these networks are called "feedforward." Recurrent neural networks are not as rigid, as backward linkages are also allowed. A recurrent connection links a neuron in a layer to either a previous layer or the neuron itself. Most recurrent neural network architectures maintain the state in the recurrent connections. Feedforward neural networks don't keep any state.

Understanding LSTM

Long Short Term Memory (LSTM) layers are a type of recurrent unit that you often use with deep neural networks.[Cite:hochreiter1997long] For TensorFlow, you can think of LSTM as a layer type that you can combine with other layer types, such as dense. LSTM makes use of two transfer function types internally.

The first type of transfer function is the sigmoid. This transfer function type is used form gates inside of the unit. The sigmoid transfer function is given by the following equation:

$$\mathrm{S}(t) = \frac{1}{1+e^{-t}}$$

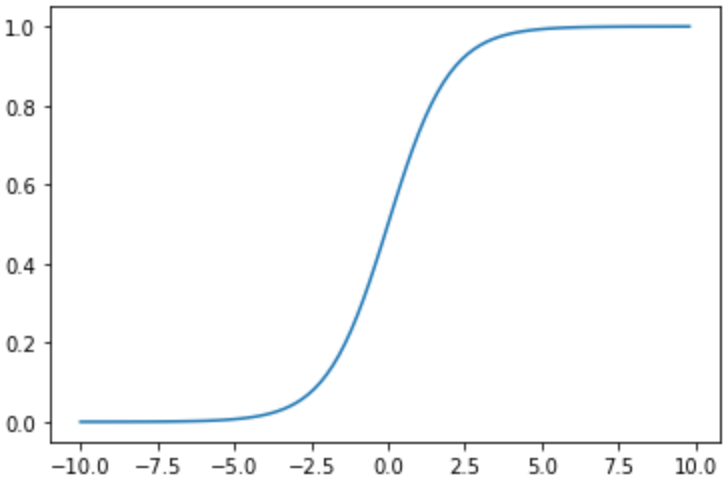
The second type of transfer function is the hyperbolic tangent (tanh) function, which allows you to scale the output of the LSTM. This functionality is similar to how we have used other transfer functions in this course.

We provide the graphs for these functions here:

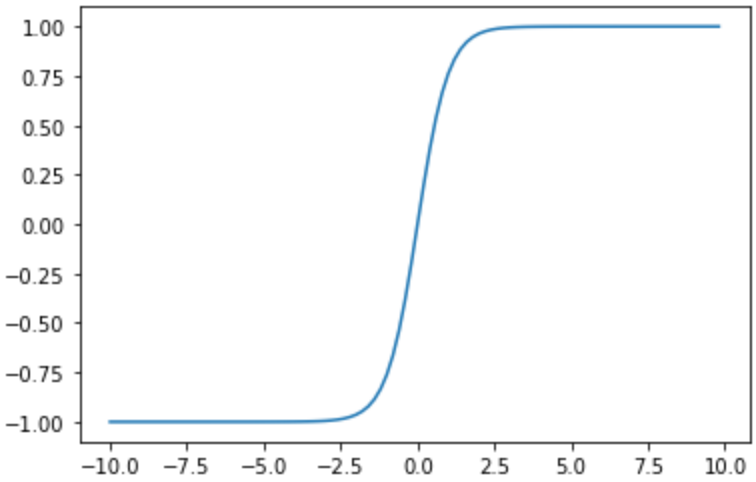
In [2]:
```
%matplotlib inline

import matplotlib
import numpy as np
import matplotlib.pyplot as plt
import math

def sigmoid(x):
    a = []
    for item in x:
        a.append(1/(1+math.exp(-item)))
    return a

def f2(x):
    a = []
    for item in x:
        a.append(math.tanh(item))
    return a
```

```
x = np.arange(-10., 10., 0.2)
y1 = sigmoid(x)
y2 = f2(x)

print("Sigmoid")
plt.plot(x,y1)
plt.show()

print("Hyperbolic Tangent(tanh)")
plt.plot(x,y2)
plt.show()
```

```
Sigmoid
```



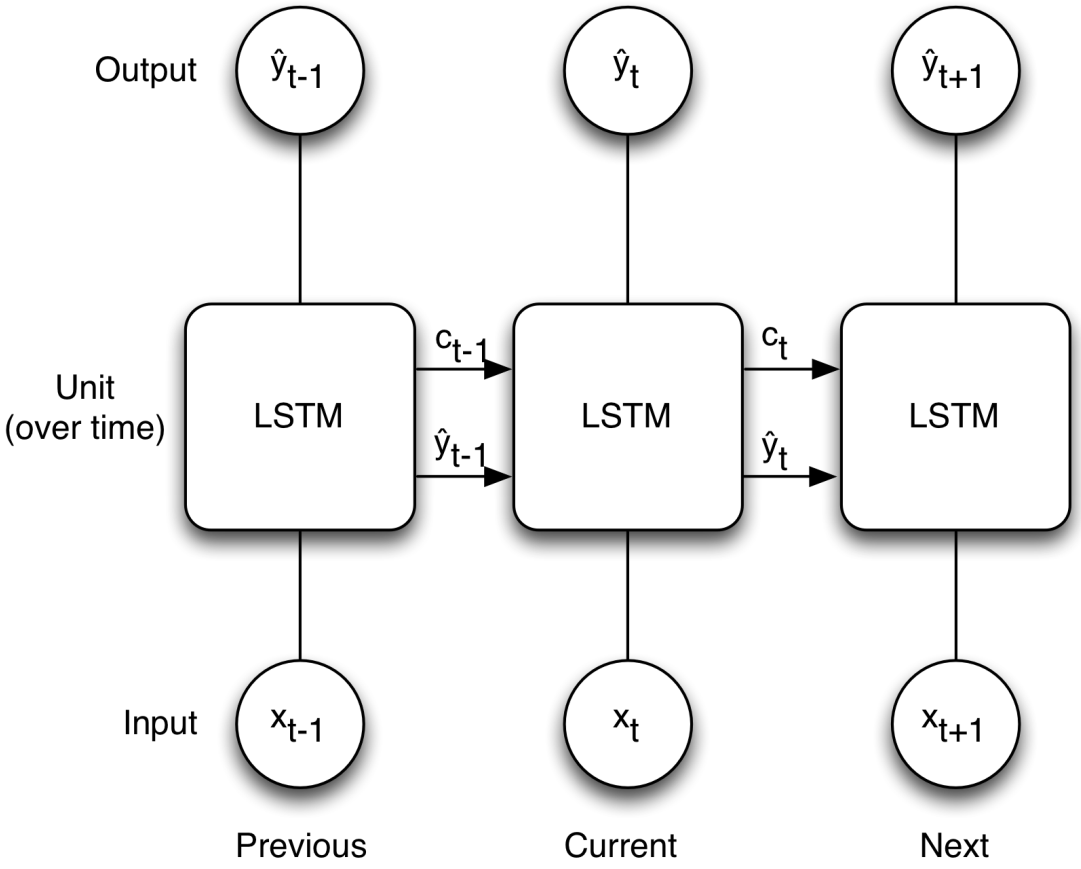

```
Hyperbolic Tangent(tanh)
```



Both of these two functions compress their output to a specific range. For the sigmoid function, this range is 0 to 1. For the hyperbolic tangent function, this range is -1 to 1.

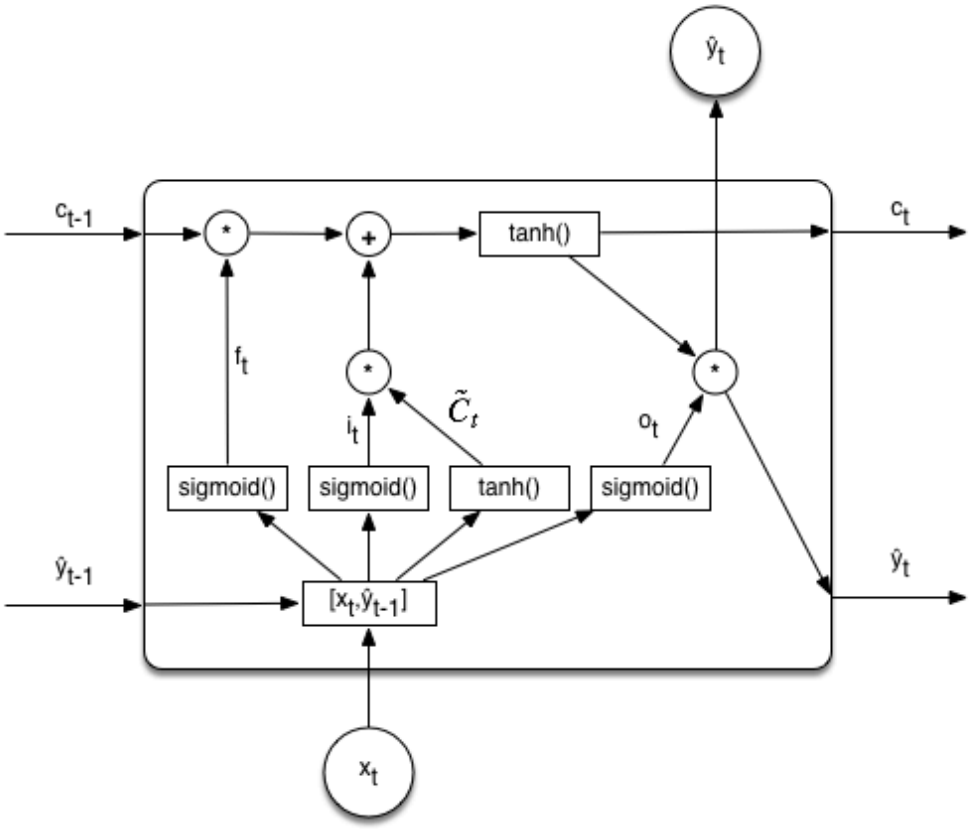
LSTM maintains an internal state and produces an output. The following diagram shows an LSTM unit over three timeslices: the current time slice (t), as well as the previous (t-1) and next (t+1) slice, as demonstrated by Figure 10.LSTM.

Figure 10.LSTM: LSTM Layers



The values $\hat{y}$ are the output from the unit; the values (x) are the input to the unit, and the values c are the context values. The output and context values always feed their output to the next time slice. The context values allow the network to maintain the state between calls. Figure 10.ILSTM shows the internals of a LSTM layer.

Figure 10.ILSTM: Inside a LSTM Layer



A LSTM unit consists of three gates:

- Forget Gate (f_t) - Controls if/when the context is forgotten. (MC)
- Input Gate (i_t) - Controls if/when the context should remember a value. (M+/MS)
- Output Gate (o_t) - Controls if/when the remembered value is allowed to pass from the unit. (RM)

Simple Keras LSTM Example

The following code creates the LSTM network, an example of an RNN for classification. The following code trains on a data set (x) with a max sequence size of 6 (columns) and six training elements (rows)

In [3]:
```
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding
from tensorflow.keras.layers import LSTM
import numpy as np

max_features = 4 # 0,1,2,3 (total of 4)
x = [
    [[0],[1],[1],[0],[0],[0]],
    [[0],[0],[0],[2],[2],[0]],
    [[0],[0],[0],[0],[3],[3]],
    [[0],[2],[2],[0],[0],[0]],
    [[0],[0],[3],[3],[0],[0]],
    [[0],[0],[0],[0],[1],[1]]
]
x = np.array(x,dtype=np.float32)
y = np.array([1,2,3,2,3,1],dtype=np.int32)

# Convert y2 to dummy variables
y2 = np.zeros((y.shape[0], max_features),dtype=np.float32)
y2[np.arange(y.shape[0]), y] = 1.0
print(y2)

print('Build model...')
model = Sequential()
model.add(LSTM(128, dropout=0.2, recurrent_dropout=0.2, \
               input_shape=(None, 1)))
model.add(Dense(4, activation='sigmoid'))

# try using different optimizers and different optimizer configs
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

print('Train...')
model.fit(x,y2,epochs=200)
pred = model.predict(x)
predict_classes = np.argmax(pred,axis=1)
print("Predicted classes: {}",predict_classes)
print("Expected classes: {}",y)
```

```
[[0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]
 [0. 1. 0. 0.]]
Build model...
Train...
Epoch 1/200
1/1 [==============================] - 6s 6s/step - loss: 0.6885 - accuracy:
0.3333
Epoch 2/200
1/1 [==============================] - 0s 45ms/step - loss: 0.6807 - accurac
y: 0.3333
Epoch 3/200
1/1 [==============================] - 0s 38ms/step - loss: 0.6773 - accurac
y: 0.3333
Epoch 4/200
1/1 [==============================] - 0s 37ms/step - loss: 0.6684 - accurac
y: 0.3333
Epoch 5/200
1/1 [==============================] - 0s 33ms/step - loss: 0.6556 - accurac
y: 0.3333
Epoch 6/200
1/1 [==============================] - 0s 37ms/step - loss: 0.6555 - accurac
y: 0.3333
Epoch 7/200
1/1 [==============================] - 0s 31ms/step - loss: 0.6397 - accurac
y: 0.1667
Epoch 8/200
1/1 [==============================] - 0s 40ms/step - loss: 0.6376 - accurac
y: 0.3333
Epoch 9/200
1/1 [==============================] - 0s 39ms/step - loss: 0.6302 - accurac
y: 0.3333
Epoch 10/200
1/1 [==============================] - 0s 39ms/step - loss: 0.6217 - accurac
y: 0.5000
Epoch 11/200
1/1 [==============================] - 0s 52ms/step - loss: 0.6196 - accurac
y: 0.6667
Epoch 12/200
1/1 [==============================] - 0s 41ms/step - loss: 0.6009 - accurac
y: 0.5000
Epoch 13/200
1/1 [==============================] - 0s 34ms/step - loss: 0.6049 - accurac
y: 0.3333
Epoch 14/200
1/1 [==============================] - 0s 42ms/step - loss: 0.5894 - accurac
y: 0.3333
Epoch 15/200
1/1 [==============================] - 0s 23ms/step - loss: 0.5853 - accurac
y: 0.3333
Epoch 16/200
1/1 [==============================] - 0s 31ms/step - loss: 0.5648 - accurac
y: 0.3333
```

```
Epoch 17/200
1/1 [==============================] - 0s 51ms/step - loss: 0.5497 - accurac
y: 0.3333
Epoch 18/200
1/1 [==============================] - 0s 39ms/step - loss: 0.5413 - accurac
y: 0.3333
Epoch 19/200
1/1 [==============================] - 0s 70ms/step - loss: 0.5429 - accurac
y: 0.3333
Epoch 20/200
1/1 [==============================] - 0s 32ms/step - loss: 0.5217 - accurac
y: 0.3333
Epoch 21/200
1/1 [==============================] - 0s 53ms/step - loss: 0.5304 - accurac
y: 0.3333
Epoch 22/200
1/1 [==============================] - 0s 58ms/step - loss: 0.5332 - accurac
y: 0.3333
Epoch 23/200
1/1 [==============================] - 0s 52ms/step - loss: 0.5090 - accurac
y: 0.3333
Epoch 24/200
1/1 [==============================] - 0s 55ms/step - loss: 0.5055 - accurac
y: 0.3333
Epoch 25/200
1/1 [==============================] - 0s 47ms/step - loss: 0.5020 - accurac
y: 0.3333
Epoch 26/200
1/1 [==============================] - 0s 47ms/step - loss: 0.5224 - accurac
y: 0.1667
Epoch 27/200
1/1 [==============================] - 0s 32ms/step - loss: 0.5060 - accurac
y: 0.1667
Epoch 28/200
1/1 [==============================] - 0s 50ms/step - loss: 0.5123 - accurac
y: 0.1667
Epoch 29/200
1/1 [==============================] - 0s 42ms/step - loss: 0.4740 - accurac
y: 0.5000
Epoch 30/200
1/1 [==============================] - 0s 40ms/step - loss: 0.5018 - accurac
y: 0.3333
Epoch 31/200
1/1 [==============================] - 0s 32ms/step - loss: 0.4982 - accurac
y: 0.3333
Epoch 32/200
1/1 [==============================] - 0s 42ms/step - loss: 0.4640 - accurac
y: 0.3333
Epoch 33/200
1/1 [==============================] - 0s 47ms/step - loss: 0.4748 - accurac
y: 0.5000
Epoch 34/200
1/1 [==============================] - 0s 50ms/step - loss: 0.4593 - accurac
y: 0.5000
Epoch 35/200
1/1 [==============================] - 0s 47ms/step - loss: 0.4712 - accurac
```

```
y: 0.3333
Epoch 36/200
1/1 [==============================] - 0s 47ms/step - loss: 0.4444 - accurac
y: 0.5000
Epoch 37/200
1/1 [==============================] - 0s 46ms/step - loss: 0.4651 - accurac
y: 0.5000
Epoch 38/200
1/1 [==============================] - 0s 41ms/step - loss: 0.4582 - accurac
y: 0.5000
Epoch 39/200
1/1 [==============================] - 0s 39ms/step - loss: 0.4191 - accurac
y: 0.5000
Epoch 40/200
1/1 [==============================] - 0s 45ms/step - loss: 0.4470 - accurac
y: 0.5000
Epoch 41/200
1/1 [==============================] - 0s 79ms/step - loss: 0.4258 - accurac
y: 0.5000
Epoch 42/200
1/1 [==============================] - 0s 35ms/step - loss: 0.4161 - accurac
y: 0.6667
Epoch 43/200
1/1 [==============================] - 0s 43ms/step - loss: 0.4094 - accurac
y: 0.6667
Epoch 44/200
1/1 [==============================] - 0s 25ms/step - loss: 0.4246 - accurac
y: 0.6667
Epoch 45/200
1/1 [==============================] - 0s 33ms/step - loss: 0.4377 - accurac
y: 0.3333
Epoch 46/200
1/1 [==============================] - 0s 37ms/step - loss: 0.4404 - accurac
y: 0.3333
Epoch 47/200
1/1 [==============================] - 0s 35ms/step - loss: 0.4526 - accurac
y: 0.3333
Epoch 48/200
1/1 [==============================] - 0s 34ms/step - loss: 0.4009 - accurac
y: 0.8333
Epoch 49/200
1/1 [==============================] - 0s 30ms/step - loss: 0.4128 - accurac
y: 0.5000
Epoch 50/200
1/1 [==============================] - 0s 39ms/step - loss: 0.3961 - accurac
y: 0.6667
Epoch 51/200
1/1 [==============================] - 0s 36ms/step - loss: 0.4152 - accurac
y: 0.6667
Epoch 52/200
1/1 [==============================] - 0s 31ms/step - loss: 0.3690 - accurac
y: 0.8333
Epoch 53/200
1/1 [==============================] - 0s 30ms/step - loss: 0.4025 - accurac
y: 0.6667
Epoch 54/200
```

```
1/1 [==============================] - 0s 40ms/step - loss: 0.3933 - accurac
y: 0.6667
Epoch 55/200
1/1 [==============================] - 0s 34ms/step - loss: 0.4072 - accurac
y: 0.8333
Epoch 56/200
1/1 [==============================] - 0s 34ms/step - loss: 0.4008 - accurac
y: 0.6667
Epoch 57/200
1/1 [==============================] - 0s 30ms/step - loss: 0.3789 - accurac
y: 0.6667
Epoch 58/200
1/1 [==============================] - 0s 51ms/step - loss: 0.3937 - accurac
y: 0.6667
Epoch 59/200
1/1 [==============================] - 0s 36ms/step - loss: 0.3516 - accurac
y: 0.6667
Epoch 60/200
1/1 [==============================] - 0s 32ms/step - loss: 0.3525 - accurac
y: 0.8333
Epoch 61/200
1/1 [==============================] - 0s 31ms/step - loss: 0.3484 - accurac
y: 1.0000
Epoch 62/200
1/1 [==============================] - 0s 53ms/step - loss: 0.3067 - accurac
y: 0.8333
Epoch 63/200
1/1 [==============================] - 0s 40ms/step - loss: 0.3600 - accurac
y: 0.5000
Epoch 64/200
1/1 [==============================] - 0s 37ms/step - loss: 0.4628 - accurac
y: 0.1667
Epoch 65/200
1/1 [==============================] - 0s 62ms/step - loss: 0.4443 - accurac
y: 0.5000
Epoch 66/200
1/1 [==============================] - 0s 33ms/step - loss: 0.3164 - accurac
y: 0.8333
Epoch 67/200
1/1 [==============================] - 0s 42ms/step - loss: 0.3303 - accurac
y: 0.8333
Epoch 68/200
1/1 [==============================] - 0s 35ms/step - loss: 0.3264 - accurac
y: 0.8333
Epoch 69/200
1/1 [==============================] - 0s 29ms/step - loss: 0.3518 - accurac
y: 0.5000
Epoch 70/200
1/1 [==============================] - 0s 27ms/step - loss: 0.2797 - accurac
y: 0.8333
Epoch 71/200
1/1 [==============================] - 0s 29ms/step - loss: 0.3595 - accurac
y: 0.5000
Epoch 72/200
1/1 [==============================] - 0s 29ms/step - loss: 0.3141 - accurac
y: 0.6667
```

```
Epoch 73/200
1/1 [==============================] - 0s 35ms/step - loss: 0.2957 - accurac
y: 0.8333
Epoch 74/200
1/1 [==============================] - 0s 33ms/step - loss: 0.3181 - accurac
y: 0.8333
Epoch 75/200
1/1 [==============================] - 0s 39ms/step - loss: 0.3049 - accurac
y: 0.6667
Epoch 76/200
1/1 [==============================] - 0s 32ms/step - loss: 0.3390 - accurac
y: 0.6667
Epoch 77/200
1/1 [==============================] - 0s 31ms/step - loss: 0.3328 - accurac
y: 0.6667
Epoch 78/200
1/1 [==============================] - 0s 41ms/step - loss: 0.3262 - accurac
y: 0.6667
Epoch 79/200
1/1 [==============================] - 0s 34ms/step - loss: 0.2353 - accurac
y: 0.8333
Epoch 80/200
1/1 [==============================] - 0s 34ms/step - loss: 0.3484 - accurac
y: 0.6667
Epoch 81/200
1/1 [==============================] - 0s 26ms/step - loss: 0.4174 - accurac
y: 0.6667
Epoch 82/200
1/1 [==============================] - 0s 32ms/step - loss: 0.2780 - accurac
y: 0.8333
Epoch 83/200
1/1 [==============================] - 0s 30ms/step - loss: 0.3406 - accurac
y: 0.5000
Epoch 84/200
1/1 [==============================] - 0s 42ms/step - loss: 0.2736 - accurac
y: 0.6667
Epoch 85/200
1/1 [==============================] - 0s 35ms/step - loss: 0.3383 - accurac
y: 0.6667
Epoch 86/200
1/1 [==============================] - 0s 51ms/step - loss: 0.3119 - accurac
y: 0.6667
Epoch 87/200
1/1 [==============================] - 0s 21ms/step - loss: 0.5429 - accurac
y: 0.3333
Epoch 88/200
1/1 [==============================] - 0s 33ms/step - loss: 0.2317 - accurac
y: 0.8333
Epoch 89/200
1/1 [==============================] - 0s 30ms/step - loss: 0.4463 - accurac
y: 0.5000
Epoch 90/200
1/1 [==============================] - 0s 39ms/step - loss: 0.3199 - accurac
y: 0.6667
Epoch 91/200
1/1 [==============================] - 0s 39ms/step - loss: 0.2997 - accurac
```

```
y: 0.8333
Epoch 92/200
1/1 [==============================] - 0s 53ms/step - loss: 0.3121 - accurac
y: 0.8333
Epoch 93/200
1/1 [==============================] - 0s 24ms/step - loss: 0.3872 - accurac
y: 0.5000
Epoch 94/200
1/1 [==============================] - 0s 30ms/step - loss: 0.3367 - accurac
y: 0.5000
Epoch 95/200
1/1 [==============================] - 0s 33ms/step - loss: 0.3248 - accurac
y: 0.6667
Epoch 96/200
1/1 [==============================] - 0s 34ms/step - loss: 0.3325 - accurac
y: 0.6667
Epoch 97/200
1/1 [==============================] - 0s 32ms/step - loss: 0.3257 - accurac
y: 0.6667
Epoch 98/200
1/1 [==============================] - 0s 33ms/step - loss: 0.3003 - accurac
y: 0.6667
Epoch 99/200
1/1 [==============================] - 0s 34ms/step - loss: 0.3220 - accurac
y: 0.6667
Epoch 100/200
1/1 [==============================] - 0s 33ms/step - loss: 0.2347 - accurac
y: 0.8333
Epoch 101/200
1/1 [==============================] - 0s 35ms/step - loss: 0.3386 - accurac
y: 0.6667
Epoch 102/200
1/1 [==============================] - 0s 30ms/step - loss: 0.3677 - accurac
y: 0.5000
Epoch 103/200
1/1 [==============================] - 0s 43ms/step - loss: 0.2826 - accurac
y: 0.6667
Epoch 104/200
1/1 [==============================] - 0s 42ms/step - loss: 0.3603 - accurac
y: 0.6667
Epoch 105/200
1/1 [==============================] - 0s 35ms/step - loss: 0.2047 - accurac
y: 0.8333
Epoch 106/200
1/1 [==============================] - 0s 36ms/step - loss: 0.2239 - accurac
y: 0.8333
Epoch 107/200
1/1 [==============================] - 0s 62ms/step - loss: 0.2280 - accurac
y: 0.8333
Epoch 108/200
1/1 [==============================] - 0s 34ms/step - loss: 0.2398 - accurac
y: 0.8333
Epoch 109/200
1/1 [==============================] - 0s 32ms/step - loss: 0.2423 - accurac
y: 0.8333
Epoch 110/200
```

```
1/1 [==============================] - 0s 25ms/step - loss: 0.2059 - accurac
y: 0.8333
Epoch 111/200
1/1 [==============================] - 0s 28ms/step - loss: 0.3120 - accurac
y: 0.6667
Epoch 112/200
1/1 [==============================] - 0s 29ms/step - loss: 0.5211 - accurac
y: 0.5000
Epoch 113/200
1/1 [==============================] - 0s 30ms/step - loss: 0.2082 - accurac
y: 0.8333
Epoch 114/200
1/1 [==============================] - 0s 47ms/step - loss: 0.2927 - accurac
y: 0.6667
Epoch 115/200
1/1 [==============================] - 0s 48ms/step - loss: 0.2020 - accurac
y: 0.8333
Epoch 116/200
1/1 [==============================] - 0s 43ms/step - loss: 0.3345 - accurac
y: 0.6667
Epoch 117/200
1/1 [==============================] - 0s 38ms/step - loss: 0.1803 - accurac
y: 0.8333
Epoch 118/200
1/1 [==============================] - 0s 43ms/step - loss: 0.1741 - accurac
y: 0.8333
Epoch 119/200
1/1 [==============================] - 0s 43ms/step - loss: 0.3439 - accurac
y: 0.5000
Epoch 120/200
1/1 [==============================] - 0s 48ms/step - loss: 0.3067 - accurac
y: 0.5000
Epoch 121/200
1/1 [==============================] - 0s 54ms/step - loss: 0.2623 - accurac
y: 0.6667
Epoch 122/200
1/1 [==============================] - 0s 38ms/step - loss: 0.2178 - accurac
y: 0.6667
Epoch 123/200
1/1 [==============================] - 0s 30ms/step - loss: 0.2032 - accurac
y: 0.8333
Epoch 124/200
1/1 [==============================] - 0s 49ms/step - loss: 0.2357 - accurac
y: 0.6667
Epoch 125/200
1/1 [==============================] - 0s 28ms/step - loss: 0.2624 - accurac
y: 0.8333
Epoch 126/200
1/1 [==============================] - 0s 41ms/step - loss: 0.2712 - accurac
y: 0.6667
Epoch 127/200
1/1 [==============================] - 0s 39ms/step - loss: 0.2747 - accurac
y: 0.6667
Epoch 128/200
1/1 [==============================] - 0s 43ms/step - loss: 0.2485 - accurac
y: 0.6667
```

```
Epoch 129/200
1/1 [==============================] - 0s 23ms/step - loss: 0.2423 - accurac
y: 0.6667
Epoch 130/200
1/1 [==============================] - 0s 36ms/step - loss: 0.2219 - accurac
y: 0.6667
Epoch 131/200
1/1 [==============================] - 0s 59ms/step - loss: 0.4338 - accurac
y: 0.6667
Epoch 132/200
1/1 [==============================] - 0s 48ms/step - loss: 0.1960 - accurac
y: 0.6667
Epoch 133/200
1/1 [==============================] - 0s 35ms/step - loss: 0.4131 - accurac
y: 0.8333
Epoch 134/200
1/1 [==============================] - 0s 37ms/step - loss: 0.2598 - accurac
y: 0.8333
Epoch 135/200
1/1 [==============================] - 0s 31ms/step - loss: 0.3850 - accurac
y: 0.6667
Epoch 136/200
1/1 [==============================] - 0s 41ms/step - loss: 0.5136 - accurac
y: 0.5000
Epoch 137/200
1/1 [==============================] - 0s 35ms/step - loss: 0.1772 - accurac
y: 0.8333
Epoch 138/200
1/1 [==============================] - 0s 30ms/step - loss: 0.3608 - accurac
y: 0.5000
Epoch 139/200
1/1 [==============================] - 0s 36ms/step - loss: 0.4406 - accurac
y: 0.6667
Epoch 140/200
1/1 [==============================] - 0s 30ms/step - loss: 0.1779 - accurac
y: 0.8333
Epoch 141/200
1/1 [==============================] - 0s 50ms/step - loss: 0.2793 - accurac
y: 0.6667
Epoch 142/200
1/1 [==============================] - 0s 35ms/step - loss: 0.4084 - accurac
y: 0.6667
Epoch 143/200
1/1 [==============================] - 0s 44ms/step - loss: 0.2198 - accurac
y: 0.8333
Epoch 144/200
1/1 [==============================] - 0s 66ms/step - loss: 0.2510 - accurac
y: 0.6667
Epoch 145/200
1/1 [==============================] - 0s 60ms/step - loss: 0.2303 - accurac
y: 0.8333
Epoch 146/200
1/1 [==============================] - 0s 46ms/step - loss: 0.1859 - accurac
y: 0.8333
Epoch 147/200
1/1 [==============================] - 0s 46ms/step - loss: 0.2167 - accurac
```

```
y: 1.0000
Epoch 148/200
1/1 [==============================] - 0s 40ms/step - loss: 0.2193 - accurac
y: 0.6667
Epoch 149/200
1/1 [==============================] - 0s 32ms/step - loss: 0.3893 - accurac
y: 0.5000
Epoch 150/200
1/1 [==============================] - 0s 43ms/step - loss: 0.1955 - accurac
y: 0.8333
Epoch 151/200
1/1 [==============================] - 0s 33ms/step - loss: 0.2113 - accurac
y: 1.0000
Epoch 152/200
1/1 [==============================] - 0s 35ms/step - loss: 0.1952 - accurac
y: 0.8333
Epoch 153/200
1/1 [==============================] - 0s 31ms/step - loss: 0.2329 - accurac
y: 0.6667
Epoch 154/200
1/1 [==============================] - 0s 41ms/step - loss: 0.2074 - accurac
y: 0.8333
Epoch 155/200
1/1 [==============================] - 0s 36ms/step - loss: 0.2839 - accurac
y: 0.6667
Epoch 156/200
1/1 [==============================] - 0s 40ms/step - loss: 0.2847 - accurac
y: 0.6667
Epoch 157/200
1/1 [==============================] - 0s 37ms/step - loss: 0.1876 - accurac
y: 0.8333
Epoch 158/200
1/1 [==============================] - 0s 51ms/step - loss: 0.3081 - accurac
y: 0.6667
Epoch 159/200
1/1 [==============================] - 0s 46ms/step - loss: 0.2110 - accurac
y: 0.8333
Epoch 160/200
1/1 [==============================] - 0s 51ms/step - loss: 0.1817 - accurac
y: 1.0000
Epoch 161/200
1/1 [==============================] - 0s 45ms/step - loss: 0.2173 - accurac
y: 0.8333
Epoch 162/200
1/1 [==============================] - 0s 51ms/step - loss: 0.1674 - accurac
y: 0.8333
Epoch 163/200
1/1 [==============================] - 0s 48ms/step - loss: 0.2233 - accurac
y: 0.6667
Epoch 164/200
1/1 [==============================] - 0s 52ms/step - loss: 0.2647 - accurac
y: 0.6667
Epoch 165/200
1/1 [==============================] - 0s 50ms/step - loss: 0.2601 - accurac
y: 0.8333
Epoch 166/200
```

```
1/1 [==============================] - 0s 43ms/step - loss: 0.2574 - accurac
y: 0.6667
Epoch 167/200
1/1 [==============================] - 0s 54ms/step - loss: 0.1954 - accurac
y: 1.0000
Epoch 168/200
1/1 [==============================] - 0s 42ms/step - loss: 0.1537 - accurac
y: 1.0000
Epoch 169/200
1/1 [==============================] - 0s 45ms/step - loss: 0.1472 - accurac
y: 1.0000
Epoch 170/200
1/1 [==============================] - 0s 52ms/step - loss: 0.2924 - accurac
y: 0.8333
Epoch 171/200
1/1 [==============================] - 0s 47ms/step - loss: 0.2536 - accurac
y: 0.8333
Epoch 172/200
1/1 [==============================] - 0s 58ms/step - loss: 0.3378 - accurac
y: 0.6667
Epoch 173/200
1/1 [==============================] - 0s 57ms/step - loss: 0.2863 - accurac
y: 0.6667
Epoch 174/200
1/1 [==============================] - 0s 49ms/step - loss: 0.2654 - accurac
y: 0.6667
Epoch 175/200
1/1 [==============================] - 0s 50ms/step - loss: 0.3798 - accurac
y: 0.6667
Epoch 176/200
1/1 [==============================] - 0s 52ms/step - loss: 0.2439 - accurac
y: 0.8333
Epoch 177/200
1/1 [==============================] - 0s 52ms/step - loss: 0.2164 - accurac
y: 0.8333
Epoch 178/200
1/1 [==============================] - 0s 42ms/step - loss: 0.2440 - accurac
y: 0.8333
Epoch 179/200
1/1 [==============================] - 0s 33ms/step - loss: 0.2581 - accurac
y: 0.6667
Epoch 180/200
1/1 [==============================] - 0s 42ms/step - loss: 0.2226 - accurac
y: 0.8333
Epoch 181/200
1/1 [==============================] - 0s 38ms/step - loss: 0.2108 - accurac
y: 0.8333
Epoch 182/200
1/1 [==============================] - 0s 51ms/step - loss: 0.1613 - accurac
y: 0.8333
Epoch 183/200
1/1 [==============================] - 0s 42ms/step - loss: 0.1548 - accurac
y: 0.8333
Epoch 184/200
1/1 [==============================] - 0s 58ms/step - loss: 0.4014 - accurac
y: 0.6667
```

```
Epoch 185/200
1/1 [==============================] - 0s 41ms/step - loss: 0.1978 - accurac
y: 1.0000
Epoch 186/200
1/1 [==============================] - 0s 63ms/step - loss: 0.3166 - accurac
y: 0.8333
Epoch 187/200
1/1 [==============================] - 0s 42ms/step - loss: 0.1714 - accurac
y: 1.0000
Epoch 188/200
1/1 [==============================] - 0s 38ms/step - loss: 0.1819 - accurac
y: 0.8333
Epoch 189/200
1/1 [==============================] - 0s 33ms/step - loss: 0.3144 - accurac
y: 0.6667
Epoch 190/200
1/1 [==============================] - 0s 35ms/step - loss: 0.2543 - accurac
y: 0.8333
Epoch 191/200
1/1 [==============================] - 0s 58ms/step - loss: 0.3698 - accurac
y: 0.6667
Epoch 192/200
1/1 [==============================] - 0s 29ms/step - loss: 0.2331 - accurac
y: 0.6667
Epoch 193/200
1/1 [==============================] - 0s 37ms/step - loss: 0.3119 - accurac
y: 0.8333
Epoch 194/200
1/1 [==============================] - 0s 43ms/step - loss: 0.2397 - accurac
y: 0.8333
Epoch 195/200
1/1 [==============================] - 0s 54ms/step - loss: 0.4169 - accurac
y: 0.6667
Epoch 196/200
1/1 [==============================] - 0s 36ms/step - loss: 0.2346 - accurac
y: 0.8333
Epoch 197/200
1/1 [==============================] - 0s 81ms/step - loss: 0.1831 - accurac
y: 0.8333
Epoch 198/200
1/1 [==============================] - 0s 45ms/step - loss: 0.2516 - accurac
y: 0.8333
Epoch 199/200
1/1 [==============================] - 0s 66ms/step - loss: 0.2622 - accurac
y: 0.6667
Epoch 200/200
1/1 [==============================] - 0s 39ms/step - loss: 0.2329 - accurac
y: 0.6667
Predicted classes: {} [1 2 3 2 3 1]
Expected classes: {} [1 2 3 2 3 1]
```

We can now present a sequence directly to the model for classification.

In [4]:
```
def runit(model, inp):
    inp = np.array(inp,dtype=np.float32)
    pred = model.predict(inp)
```

```
    return np.argmax(pred[0])

print( runit( model, [[[0],[0],[0],[0],[0],[1]]] ))
```

```
1
```

Sun Spots Example

This section shows an example of RNN regression to predict sunspots. You can find the data files needed for this example at the following location.

- Sunspot Data Files
- Download Daily Sunspots - 1/1/1818 to now.

The following code loads the sunspot file:

In [5]:
```
import pandas as pd
import os

names = ['year', 'month', 'day', 'dec_year', 'sn_value' ,
         'sn_error', 'obs_num', 'unused1']
df = pd.read_csv(
    "https://data.heatonresearch.com/data/t81-558/SN_d_tot_V2.0.csv",
    sep=';',header=None,names=names,
    na_values=['-1'], index_col=False)

print("Starting file:")
print(df[0:10])

print("Ending file:")
print(df[-10:])
```

```
Starting file:
   year  month  day  dec_year  sn_value  sn_error  obs_num  unused1
0  1818      1    1  1818.001        -1       NaN        0        1
1  1818      1    2  1818.004        -1       NaN        0        1
2  1818      1    3  1818.007        -1       NaN        0        1
3  1818      1    4  1818.010        -1       NaN        0        1
4  1818      1    5  1818.012        -1       NaN        0        1
5  1818      1    6  1818.015        -1       NaN        0        1
6  1818      1    7  1818.018        -1       NaN        0        1
7  1818      1    8  1818.021        65      10.2        1        1
8  1818      1    9  1818.023        -1       NaN        0        1
9  1818      1   10  1818.026        -1       NaN        0        1
Ending file:
       year  month  day  dec_year  sn_value  sn_error  obs_num  unused1
72855  2017      6   21  2017.470        35       1.0       41        0
72856  2017      6   22  2017.473        24       0.8       39        0
72857  2017      6   23  2017.475        23       0.9       40        0
72858  2017      6   24  2017.478        26       2.3       15        0
72859  2017      6   25  2017.481        17       1.0       18        0
72860  2017      6   26  2017.484        21       1.1       25        0
72861  2017      6   27  2017.486        19       1.2       36        0
72862  2017      6   28  2017.489        17       1.1       22        0
72863  2017      6   29  2017.492        12       0.5       25        0
72864  2017      6   30  2017.495        11       0.5       30        0
```

As you can see, there is quite a bit of missing data near the end of the file. We want to find the starting index where the missing data no longer occurs. This technique is somewhat sloppy; it would be better to find a use for the data between missing values. However, the point of this example is to show how to use LSTM with a somewhat simple time-series.

In [6]:
```
start_id = max(df[df['obs_num'] == 0].index.tolist())+1  # Find the last zer
print(start_id)
df = df[start_id:] # Trim the rows that have missing observations
```

```
11314
```

In [7]:
```
df['sn_value'] = df['sn_value'].astype(float)
df_train = df[df['year']<2000]
df_test = df[df['year']>=2000]

spots_train = df_train['sn_value'].tolist()
spots_test = df_test['sn_value'].tolist()

print("Training set has {} observations.".format(len(spots_train)))
print("Test set has {} observations.".format(len(spots_test)))
```

```
Training set has 55160 observations.
Test set has 6391 observations.
```

To create an algorithm that will predict future values, we need to consider how to encode this data to be presented to the algorithm. The data must be submitted as sequences, using a sliding window algorithm to encode the data. We must define how large the window will be. Consider an n-sized window. Each sequence's x values will be

a n data points sequence. The y's will be the next value, after the sequence, that we are trying to predict. You can use the following function to take a series of values, such as sunspots, and generate sequences (x) and predicted values (y).

In [8]:
```
import numpy as np

def to_sequences(seq_size, obs):
    x = []
    y = []

    for i in range(len(obs)-SEQUENCE_SIZE):
        #print(i)
        window = obs[i:(i+SEQUENCE_SIZE)]
        after_window = obs[i+SEQUENCE_SIZE]
        window = [[x] for x in window]
        #print("{} - {}".format(window,after_window))
        x.append(window)
        y.append(after_window)

    return np.array(x),np.array(y)


SEQUENCE_SIZE = 10
x_train,y_train = to_sequences(SEQUENCE_SIZE,spots_train)
x_test,y_test = to_sequences(SEQUENCE_SIZE,spots_test)

print("Shape of training set: {}".format(x_train.shape))
print("Shape of test set: {}".format(x_test.shape))
```

```
Shape of training set: (55150, 10, 1)
Shape of test set: (6381, 10, 1)
```

We can see the internal structure of the training data. The first dimension is the number of training elements, the second indicates a sequence size of 10, and finally, we have one data point per timeslice in the window.

In [9]:
```
x_train.shape
```

Out[9]:
```
(55150, 10, 1)
```

We are now ready to build and train the model.

In [ ]:
```
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding
from tensorflow.keras.layers import LSTM
from tensorflow.keras.datasets import imdb
from tensorflow.keras.callbacks import EarlyStopping
import numpy as np

print('Build model...')
model = Sequential()
model.add(LSTM(64, dropout=0.0, recurrent_dropout=0.0,\
                   input_shape=(None, 1)))
```

```
model.add(Dense(32))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
monitor = EarlyStopping(monitor='val_loss', min_delta=1e-3, patience=5,
                        verbose=1, mode='auto', restore_best_weights=True)
print('Train...')

model.fit(x_train,y_train,validation_data=(x_test,y_test),
          callbacks=[monitor],verbose=2,epochs=1000)
```

```
Build model...
Train...
Epoch 1/1000
1724/1724 - 12s - loss: 1392.9260 - val_loss: 212.4303 - 12s/epoch - 7ms/ste
p
Epoch 2/1000
1724/1724 - 10s - loss: 516.7653 - val_loss: 239.7139 - 10s/epoch - 6ms/step
Epoch 3/1000
1724/1724 - 10s - loss: 509.0716 - val_loss: 227.2393 - 10s/epoch - 6ms/step
Epoch 4/1000
1724/1724 - 10s - loss: 503.2472 - val_loss: 257.9693 - 10s/epoch - 6ms/step
Epoch 5/1000
1724/1724 - 10s - loss: 498.1863 - val_loss: 226.4870 - 10s/epoch - 6ms/step
Epoch 6/1000
1724/1724 - 10s - loss: 499.1180 - val_loss: 208.2820 - 10s/epoch - 6ms/step
Epoch 7/1000
1724/1724 - 10s - loss: 498.4828 - val_loss: 224.2135 - 10s/epoch - 6ms/step
Epoch 8/1000
1724/1724 - 10s - loss: 497.6813 - val_loss: 253.0776 - 10s/epoch - 6ms/step
Epoch 9/1000
1724/1724 - 10s - loss: 496.7933 - val_loss: 211.7351 - 10s/epoch - 6ms/step
Epoch 10/1000
1724/1724 - 10s - loss: 497.0393 - val_loss: 215.1721 - 10s/epoch - 6ms/step
Epoch 11/1000
Restoring model weights from the end of the best epoch: 6.
1724/1724 - 10s - loss: 495.1920 - val_loss: 220.1826 - 10s/epoch - 6ms/step
Epoch 11: early stopping
```

Finally, we evaluate the model with RMSE.

In [ ]:
```
from sklearn import metrics

pred = model.predict(x_test)
score = np.sqrt(metrics.mean_squared_error(pred,y_test))
print("Score (RMSE): {}".format(score))
```