Language: Python3.8
Framework: PyTorch
Libraries: Numpy, Pandas, PyTorch, Matplotlib, Sklearn
Environment: Jupyter Notebook

Task:
Develop **LSTM** for any sequence prediction

Outcome:
Trained LSTM to predict 1000 points for a sine wave

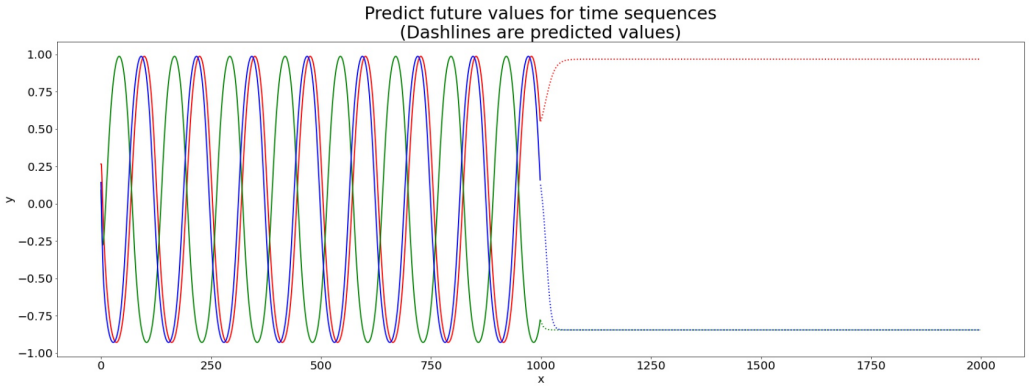
Model:

```
class Sequence(nn.Module):
   hidden_size = 10
   def __init__(self):
        super(Sequence, self).__init__()
        self.lstm1 = nn.LSTMCell(1, self.hidden_size)
        self.lstm2 = nn.LSTMCell(self.hidden_size, self.hidden_size)
        self.linear = nn.Linear(self.hidden_size, 1)

   def forward(self, input, future = 0):
        outputs = []
        h_t = torch.zeros(input.size(0), self.hidden_size,
dtype=torch.double)
        c_t = torch.zeros(input.size(0), self.hidden_size,
dtype=torch.double)
        h_t2 = torch.zeros(input.size(0), self.hidden_size,
dtype=torch.double)
        c_t2 = torch.zeros(input.size(0), self.hidden_size,
dtype=torch.double)

        for input_t in input.split(1, dim=1):
            h_t, c_t = self.lstm1(input_t, (h_t, c_t))
            h_t2, c_t2 = self.lstm2(h_t, (h_t2, c_t2))
            output = self.linear(h_t2)
            outputs += [output]
        for i in range(future):# if we should predict the future
            h_t, c_t = self.lstm1(output, (h_t, c_t))
            h_t2, c_t2 = self.lstm2(h_t, (h_t2, c_t2))
            output = self.linear(h_t2)
```

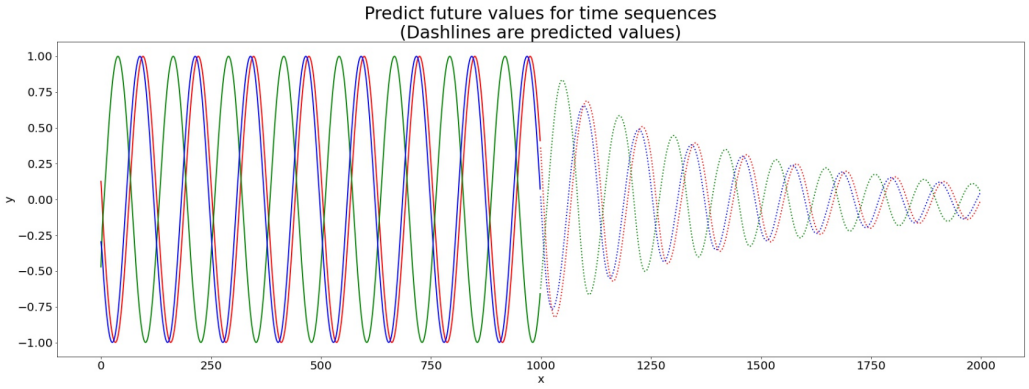
```
        outputs += [output]
    outputs = torch.cat(outputs, dim=1)
    return outputs
```

Results:

Iteration 1:



Iteration 5:



Iteration 15 (Final):