


torch.nn.LSTM()详解

原创xhsun1997 于 2020-02-23 11:43:53 发布 43509 收藏 240 版权

分类专栏：tensorflow+pytorch 文章标签：神经网络 python 深度学习 算法

 tensorflow+pytorch 专栏收录该内容

3 订阅 9 篇文章 订阅专栏

输入的参数列表包括:

- input_size 输入数据的特征维数，通常就是embedding_dim(词向量的维度)
- hidden_size LSTM中隐层的维度
- num_layers 循环神经网络的层数
- bias 用不用偏置，default=True
- batch_first 这个要注意，通常我们输入的数据shape=(batch_size,seq_length,embedding_dim),而batch_first默认是False,所以我们的输入数据最好送进LSTM之前将batch_size与seq_length这两个维度调换
- dropout 默认是0，代表不用dropout
- bidirectional默认是false，代表不用双向LSTM

输入数据包括input,(h_0,c_0):

- input就是shape=(seq_length,batch_size,input_size)的张量
- h_0是shape=(num_layers*num_directions,batch_size,hidden_size)的张量，它包含了在当前这个batch_size中每个句子的初始隐藏状态。其中num_layers就是LSTM的层数。如果bidirectional=True,num_directions=2,否则就是1，表示只有一个方向。
- c_0和h_0的形状相同，它包含的是在当前这个batch_size中的每个句子的初始细胞状态。h_0,c_0如果不提供，那么默认是0。

输出数据包括output,(h_n,c_n):

- output的shape=(seq_length,batch_size,num_directions*hidden_size),它包含的是LSTM的最后一时间步的输出特征(h_t), t 是batch_size中每个句子的长度。
- h_n.shape==(num_directions * num_layers,batch,hidden_size)
- c_n.shape==h_n.shape
- h_n包含的是句子的最后一个单词（也就是最后一个时间步）的隐藏状态，c_n包含的是句子的最后一个单词的细胞状态，所以它们都与句子的长度seq_length无关。
- output[-1]与h_n是相等的，因为output[-1]包含的正是batch_size个句子中每一个句子的最后一个单词的隐藏状态，注意LSTM中的隐藏状态其实就是输出，cell state细胞状态才是LSTM中一直隐藏的，记录着信息

```
1 import torch
2 batch_size=3
3 hidden_size=5
4 embedding_dim=6
5 seq_length=4
6 num_layers=1
7 num_directions=1
8 vocab_size=20
9 import numpy as np
10 input_data=np.random.uniform(0,19,size=(batch_size,seq_length))
11 input_data=torch.from_numpy(input_data).long()
12 embedding_layer=torch.nn.Embedding(vocab_size,embedding_dim)
13 lstm_layer=torch.nn.LSTM(input_size=embedding_dim,hidden_size=hidden_size,num_layers=num_layers,
14                           bias=True,batch_first=False,dropout=0.5,bidirectional=False)
15 lstm_input=embedding_layer(input_data)
16 assert lstm_input.shape==(batch_size,seq_length,embedding_dim)
17 lstm_input.transpose_(1,0)
18 assert lstm_input.shape==(seq_length,batch_size,embedding_dim)
19 output,(h_n,c_n)=lstm_layer(lstm_input)
20 assert output.shape==(seq_length,batch_size,hidden_size)
21 assert h_n.shape==c_n.shape==(num_layers*num_directions,batch_size,hidden_size)
```

1	h_n
tensor([[[0.0974, -0.0158, -0.1922, -0.2420, 0.0570], [-0.1126, -0.0290, -0.1625, 0.1130, 0.0874], [-0.0859, -0.1010, 0.0291, 0.0195, 0.1560]]], grad_fn=<StackBackward>)	

1	output[-1]
tensor([[0.0974, -0.0158, -0.1922, -0.2420, 0.0570], [-0.1126, -0.0290, -0.1625, 0.1130, 0.0874], [-0.0859, -0.1010, 0.0291, 0.0195, 0.1560]], grad_fn=<SelectBackward>)	

https://blog.csdn.net/m0_45478865



创作打卡挑战
赢取流量/现金/C



xhsun1997

关注

64



14



240



专栏目录

扫一扫，分享内容



点击复制链接