

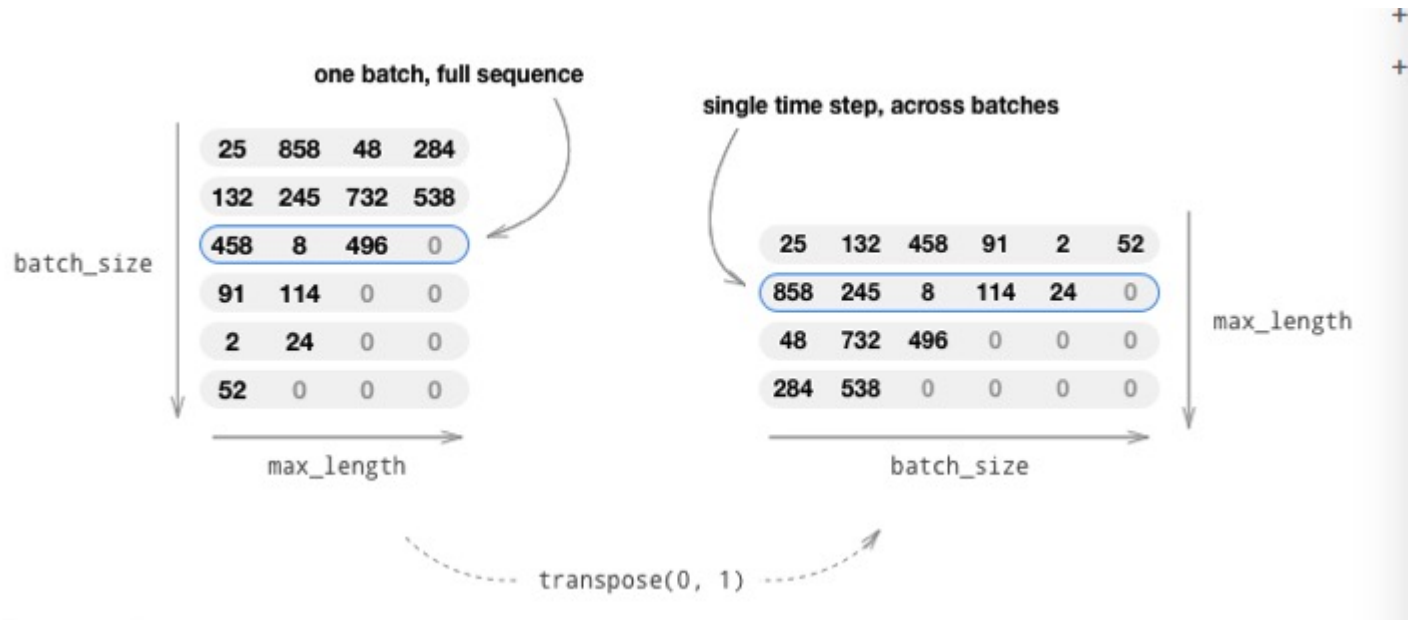
## lstm pytorch梳理之 batch\_first 参数 和torch.nn.utils.rnn.pack\_padded\_sequence

小萌新在看pytorch官网 LSTM代码时 对batch\_first 参数 和torch.nn.utils.rnn.pack\_padded\_sequence 不太理解,

在回去苦学了一番 , 将自己消化过的记录在这 , 希望能帮到跟我有同样迷惑的伙伴

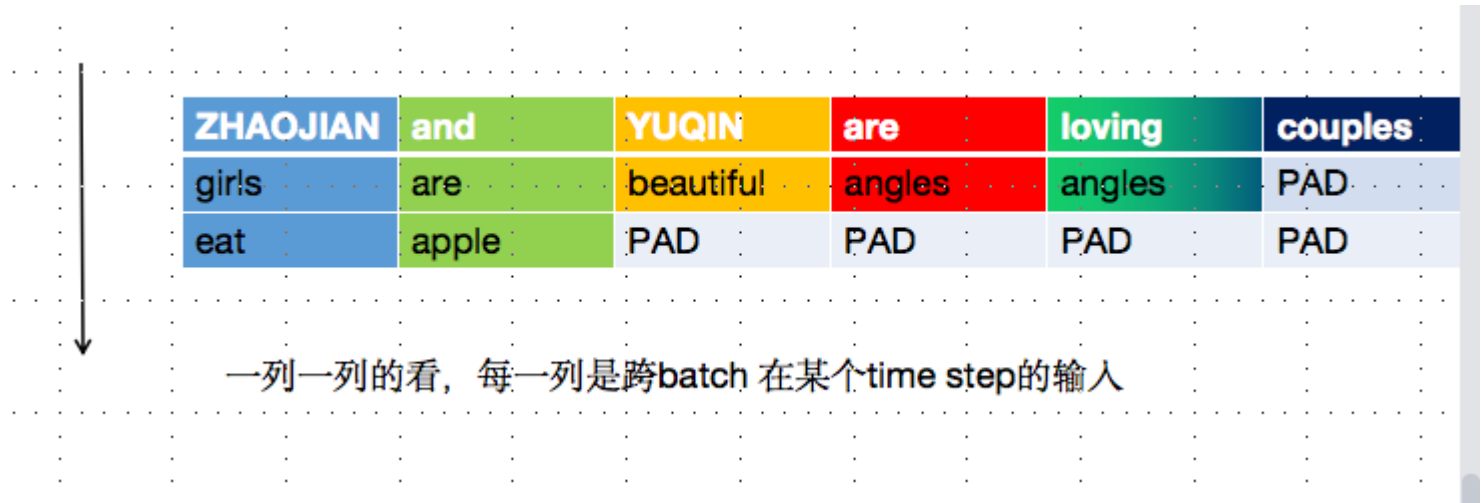
官方API: <https://pytorch.org/docs/stable/nn.html?highlight=lstm#torch.nn.LSTM>

- 参数
  - input\_size
  - hidden\_size
  - num\_layers
  - bias
  - batch\_first
  - dropout
  - bidirectional
- 特别说下batch\_first ,参数默认为False, 也就是它鼓励我们第一维不是batch, 这与我们常规输入想悖, 毕竟我们习惯的输入是(batch, seq\_len, hidden\_size),那么官方为啥会 这样子设置呢?



先不考虑hiddem\_dim,左边图矩阵维度为batch\_size \* max\_length, 6个序列, 没个序列填充到最大长度4, 经过转置后得到max\_length \* batch\_size, 右图标蓝的一列 对应的就是 左图第二列, 而左图第二列表示的是 每个序列里面第二个token, 这样子有什么好处呢? 相当于可以并行处理 每个句子在time step下时刻的计算, 这样就可**以并行过LSTM, 从而一定程度上提高处理速度**. 因为官网放的图例子里面数字都是句子token 索引化之后的, 反而让人容易看晕, 因而小萌新自己画了个好理解的图。一起看下图呀。

一共有3个句子, 最大长度为6, 我们之前习惯的是 按行看, 我们现在按一列一列来看 (就相当于转置啦)



time step 0接受的是[ZHAOJIAN girls eat];

time step1接收的是[and are apple];

time step2接受的是[YUQIN beautiful PAD];

time step3接收的是[are angles PAD];

以此类推。 现在这3个句子就可以**并行过LSTM**

### pad\_sequence

我们知道一个batch里的序列长度是不一致的, 而LSTM是无法处理长度不同的序列的, 需要pad操作用0把它们都填充成max\_length 长度。下图有3个句子, 以最长的句子长度 6 作为max\_length, 其余句子都填充到max\_length 。这是PAD的作用, 很好理解。

```
from torch.nn.utils.rnn import pack_padded_sequence ,pad_sequence ,pack_sequence

inputs = ["LIHUA went to The Tsinghua University",
          "Liping went to technical school ",
          "I work in the mall ",
          "we both have bright future"]

inputs.sort(key=lambda x:len(x.split()),reverse=True)
batch_size=len(inputs)
max_length=len(inputs[0].split())
lengths=[len(s.split()) for s in inputs]

word_to_idx={}
for sen in inputs:
    for word in sen.split():
        if word not in word_to_idx:
            word_to_idx[word]=len(word_to_idx)

idx=[]
for sentence in inputs:
    a=[word_to_idx[w] for w in sentence.split()]
    idx.append(a)

pprint(idx)
padded_sequence = pad_sequence([torch.FloatTensor(id) for id in idx], batch_first=True)
```

公告

昵称: [打了鸡血的女汉子](#)  
园龄: [1年5个月](#)  
粉丝: [9](#)  
关注: [0](#)  
[+加关注](#)

<	2022年5月						>
日	一	二	三	四	五	六	
1	2	3	4	5	6	7	
8	9	10	11	12	13	14	
15	16	17	18	19	20	21	
22	23	24	25	26	27	28	
29	30	31	1	2	3	4	
5	6	7	8	9	10	11	

搜索

常用链接

[我的随笔](#)  
[我的评论](#)  
[我的参与](#)  
[最新评论](#)  
[我的标签](#)

随笔分类

[pytorch学习地图\(3\)](#)  
[平平无奇读paper小能手\(2\)](#)  
[文档搜索\(4\)](#)

随笔档案

[2021年1月\(1\)](#)  
[2020年12月\(16\)](#)

阅读排行榜

- [1. Embedding模块 from\\_pretrained 加载预训练好的词向量 \(4488\)](#)
- [2. lstm pytorch梳理之 batch\\_first 参数 和torch.n n.utils.rnn.pack\\_padded\\_sequence\(2110\)](#)
- [3. query-doc 匹配\(1217\)](#)
- [4. bilstm+crf pytorch 代码 保姆式吐血整理\(807\)](#)
- [5. pytorch transformers finetune 疑惑总结\(678\)](#)

评论排行榜

- [1. 交作业 之 pytorch 使用字符级特征来增强 LSTM 词性标注器\(3\)](#)
- [2. 读论文啦! Context-Aware Sentence/Passage Term Importance Estimation For First Stage Retrieval\(2\)](#)
- [3. 读论文啦! 相关性匹配经典论文A Deep Relevance Matching Model for Ad-hoc Retrieval\(2\)](#)
- [4. lstm pytorch梳理之 batch\\_first 参数 和torch.n n.utils.rnn.pack\\_padded\\_sequence\(2\)](#)
- [5. bilstm+crf pytorch 代码 保姆式吐血整理\(2\)](#)

推荐排行榜

- [1. 交作业 之 pytorch 使用字符级特征来增强 LSTM 词性标注器\(5\)](#)
- [2. lstm pytorch梳理之 batch\\_first 参数 和torch.n n.utils.rnn.pack\\_padded\\_sequence\(4\)](#)
- [3. 读论文啦! Context-Aware Sentence/Passage Term Importance Estimation For First Stage Retrieval\(2\)](#)
- [4. 读论文啦! 相关性匹配经典论文A Deep Relevance Matching Model for Ad-hoc Retrieval\(2\)](#)
- [5. 读论文啦! Bridging the Gap Between Relevance Matching and Semantic Matching for Short Text Similarity Modeling\(2\)](#)

最新评论

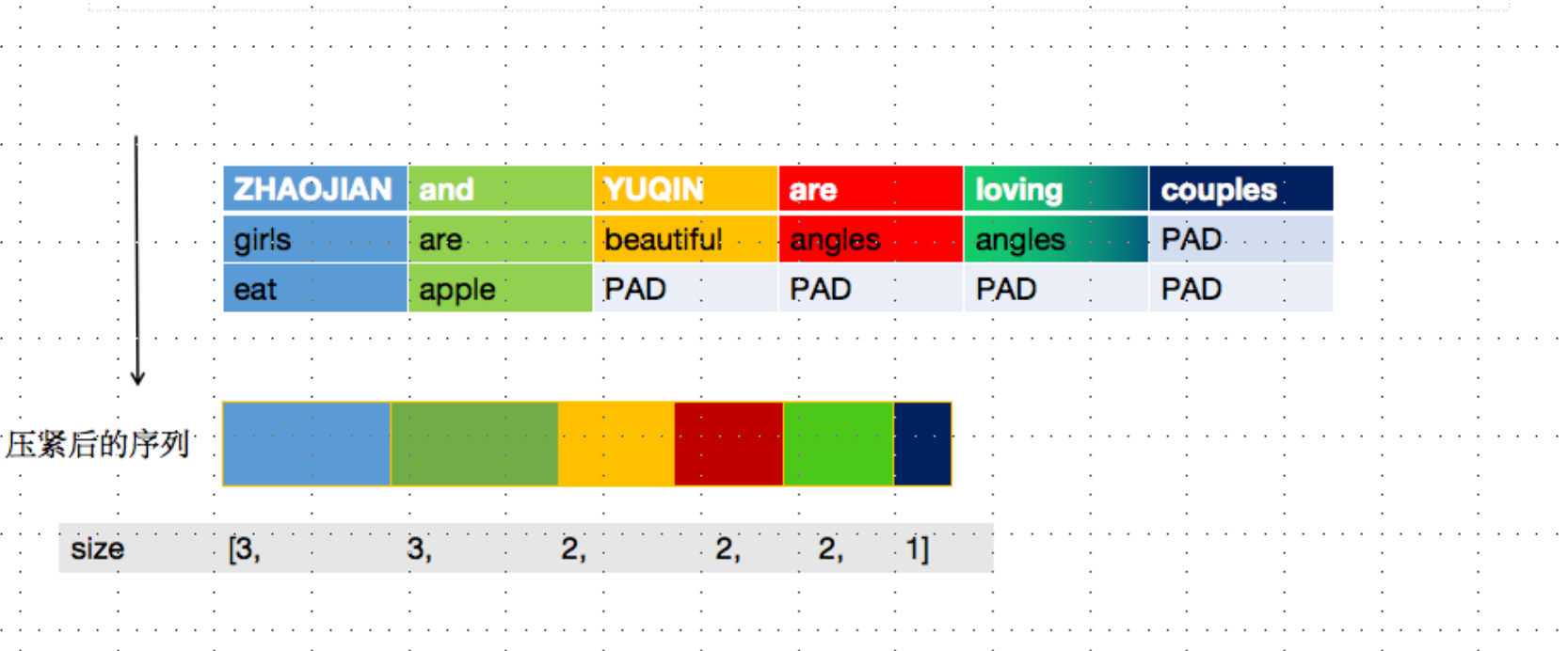
- [1. Re:bilstm+crf pytorch 代码 保姆式吐血整理](#)  
这个 lstm+CRF 的代码哪里有呀?  
--吼吼, 哈哈, 啦啦
- [2. Re:lstm pytorch梳理之 batch\\_first 参数 和 torch.nn.utils.rnn.pack\\_padded\\_sequence](#)  
@Harukaze 哈哈哈哈哈谢谢对我的鼓励...  
--打了鸡血的女汉子
- [3. Re:lstm pytorch梳理之 batch\\_first 参数 和 torch.nn.utils.rnn.pack\\_padded\\_sequence](#)  
很棒, 其他文章也很棒, 抽时间学习  
--Harukaze
- [4. Re:交作业 之 pytorch 使用字符级特征来增强 LSTM 词性标注器](#)  
讲解的很详细  
--surreal
- [5. Re:交作业 之 pytorch 使用字符级特征来增强 LSTM 词性标注器](#)  
论文分析的挺好的  
--surreal

```
print(padded_sequence)
packed_sequence = pack_sequence([torch.FloatTensor(id) for id in idx]) # packed_sequence是PackedSequence的实例
```

pack\_sequence

但带来一个问题，什么问题呢？对于长度小于MAX\_LENGTH，经过PAD填充操作后的句子，会导致LSTM对它的表示多了很多无用的字符，如下图所示，我们希望的是在最后一个有用token 就输入句子的向量表示，而不是在很多PAD后才输入句子表示，这是pack就派上场了，可以理解成 将一个填充过的变长序列压紧.压缩的对象就是 padded suquence, 压缩后的输入将不含 0

看图更好理解 哦



那么，聪明如你肯定会觉得不对劲，这先 填充又 压紧, 这不是做无用功？其实不是哦，因为 pack 后可并不是一个简单的 Tensor 类型的数据，而是一个 “PackedSequence” 类型的 object，可以直接传给RNN。小萌新在苦逼地看 RNN源码时，发现forward 函数 里上来就是判断 输入是否是 PackedSequence 的实例，进而采取不同的操作。如果输入是 PackedSequence，输出也是该类型。这里的输出类型都指的是 forward 函数的第一个返回值（每个time step 对应的hidden\_state），第二个返回值(最后一个time step对应的hidden\_state)的类型不管输入是不是 PackedSequence 类型，都是一样的。

pack\_padded\_sequence

pytorch里 有封装的更好的：torch.nn.utils.rnn.pack\_padded\_sequence(input, lengths, batch\_first=False, enforce\_sorted=True)

接下来说说这些参数作用

lengths：该参数中各句子长度值的顺序要和对应的输入中的序列顺序一致

enforce\_sorted：默认值是 True，表示输入已经按句子长度降序排好序。如果输入在 pad 时没有顺序，那么此时在此处需要设置该值为 False,那么函数会再去排序

返回的对象是PackedSequence object。该类型的变量便可以直接喂给 RNN/LSTM等。

torch.nn.utils.rnn.pad\_packed\_sequence():之前的pack\_padded\_sequence 是先补齐到相同长度 再压紧，这个当然就是反过来，对压紧后的序列 进行扩充补齐操作。

注意: inputs是否排好序和 lengths参数和enforce\_sorted 一定要对应起来。小萌新习惯将 inputs 按照长度先排好序，再将length 排好序enforce\_sorted参数不去动它。

```
inputs.sort(key=lambda x:len(x.split()),reverse=True)
lengths=[len(s.split()) for s in inputs]
```

```
def forward(self, input_seq, input_lengths, hidden=None):
    # Convert word indexes to embeddings
    embedded = self.embedding(input_seq)
    # Pack padded batch of sequences for RNN module
    packed = nn.utils.rnn.pack_padded_sequence(embedded, input_lengths)
    # Forward pass through GRU
    outputs, hidden = self.gru(packed, hidden)
    # Unpack padding
    outputs, _ = nn.utils.rnn.pad_packed_sequence(outputs)
    # Sum bidirectional GRU outputs
    outputs = outputs[:, :, :self.hidden_size] + outputs[:, :, self.hidden_size:]
    # Return output and final hidden state
    return outputs, hidden
```

-----恢复内容结束-----

分类: [pytorch学习地图](#)

好文置顶

关注我

收藏该文

打了鸡血的女汉子

关注 - 0

粉丝 - 9

+加关注

« 上一篇: [bilstm+ctf 吐血的理论讲解](#)

» 下一篇: [pycharm 修改文件名-Refactor](#)