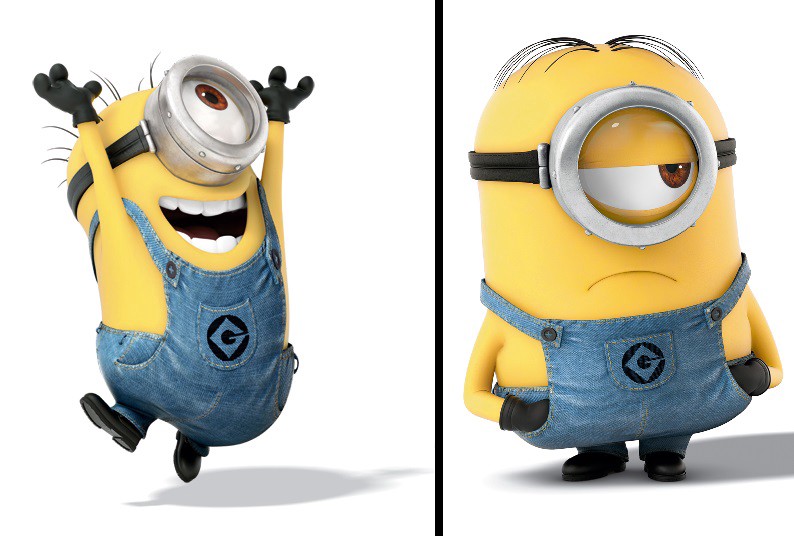
# 情绪分析



这篇文章是本系列的第二部分。在纯pytorch中建立了情感分析模型。在这篇文章中，我做了同样的工作，但是在和演示了torchtext的优点，并且它减少了很多代码。

情绪分析是一项分类任务，每个样本都有一个正面或负面的标签。你可以跟着代码走。

从[2]下载数据集

#### NLP中分类任务的典型组成

1. 预处理和标记化
2. 生成唯一标记的词汇表并将单词转换为索引
3. 加载预训练向量，如手套、Word2vec、Fasttext
4. 在长度可变的情况下用零填充文本
5. 数据加载和批处理
6. 模型创建和培训

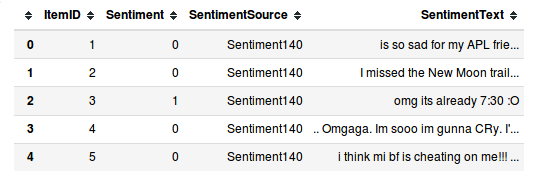
#### 为什么使用torchtext

Torchtext提供了一组在NLP任务中有用的类。这些类用最少的代码处理上面的前5点。我们将详细研究每一点。

我已经在train和validation set中分割数据并保存为csv。

注意：在将csv保存为torchtext之前，请确保删除所有“”字符，否则无法处理“”字符。

#### 一。定义如何处理数据



第一步是声明我们要在数据帧中使用哪些属性（列）以及如何处理它们。数据框架由4列组成（“ItemID”、“mousion”、“mousionsource”、“mousiontext”），我们只想使用“mousion”和“mousiontext”。

label列（'mousion'）是二进制的，并且已经是数字形式，因此不需要处理它。tweet列（'momentText'）需要处理和标记化，以便可以将其转换为索引。

让我们把上面的代码分解一下。在torchtext中，列可以称为字段。field对象接受关于如何处理（标记化等）文本的参数。此字段对象稍后将附加到数据集。

第10行定义了将来传递实际数据（tweets）时如何处理列或字段的蓝图。

第15行定义了将来传递实际数据（标签）时如何处理列或字段的蓝图。

在第20行中，我们定义了如何处理每一列。没有值的列将被忽略并且不会被加载。

#### 2。创建torchtext数据集

您可能会问为什么不使用默认的pytorch数据集。原因是torchtext提供了一组专门用于NLP任务的数据集。其中一个这样的数据集是TabularDataset，专门用来读取csv和tsv文件并对其进行处理。它是pytorch数据集的包装器，具有其他特性。

到目前为止，我们已经确定了加工的蓝图。现在我们将实际加载数据进行处理。

第27行包含TabularDataset.splits（）方法用于一次性处理多个使用相同处理的文件（火车、验证、测试）。

splits（）将返回火车数据集和验证数据集。

让我们看看这个TabularDataset对象包含什么。

TabularDataset是一个包含示例对象的列表。示例对象在单个对象中包装所有列（文本和标签）。这些列可以通过在上述代码中写入的列名进行访问。

#### 三。加载预训练的词向量并构建词汇

Torchtext使得加载预先训练的词向量变得非常容易。只要提到预训练的单词向量的名称（例如glove.6B.50d、fasttext.en.300d等），torchtext就会下载该向量，然后可以在嵌入层中使用它。

或者，如果已经下载了预训练向量，则可以指定路径，torchtext将读取并加载该路径。我在下面的代码中使用了下载的向量。

在上面的第4行和第6行中，torchtext基于“感伤文本”列中提供的文本构建词汇表词汇表是建立在列车中的文本数据集和验证数据集上的，并将最大唯一单词数定义为100000个。不在词汇表中的单词将被分配令牌。在词汇构建过程中传递预训练向量。

现在，当您执行第4行时，torchtext将创建一个包含所有唯一单词的字典，并按其频率的降序排列它们，并在此字典的开头添加和标记。下一个torchtext为每个单词分配唯一的整数，并将此映射保存在txt-field.vocab.stoi（字符串到索引）中，并将此映射保留在txt-field.vocab.itos（索引到字符串）中。

#### 四。批量加载数据

对于具有可变长度语句的数据，torchtext提供BucketIterator（）dataloader，它是pytorch dataloader的包装器。BucketIterator提供了一些额外的好处，比如根据文本的长度对数据进行排序，并在批处理中将相似长度的文本组合在一起。这有助于减少所需的填充量。请注意，我没有将文本固定到任何特定长度。根据最大长度的样品垫批。

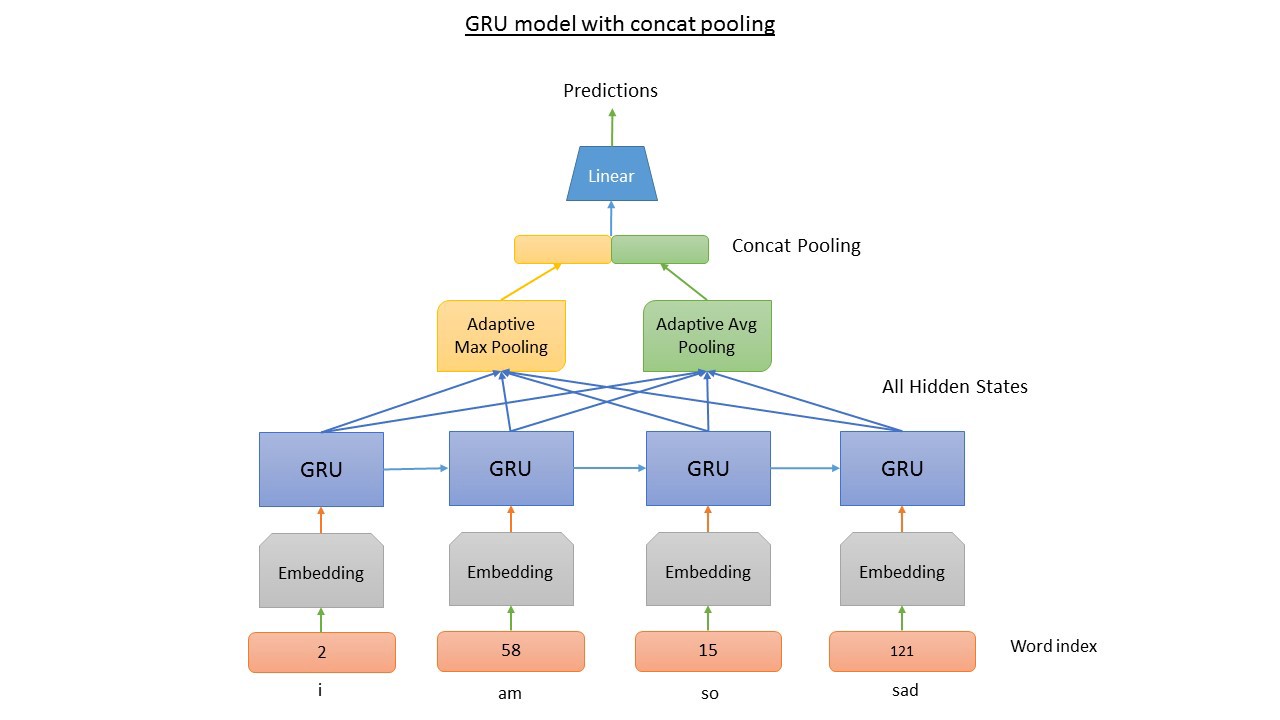
注意：BucketIterator返回一个批处理对象，而不是文本索引和标签。批处理对象也不像pytorch Dataloader那样是不可iterable的。单个批对象包含一个批的数据。可以通过列名访问文本和标签。

这是torchtext中的一个小问题。但这可以通过两种方式轻松克服。在训练循环中编写一些额外的代码，以便从批处理对象中获取数据，或者在批处理对象周围编写一个返回所需数据的iterable包装器。我将采取第二种方法，因为这是更干净的。

通过上面的代码，我们可以直接在训练循环中使用它，就像pytorch Dataloader一样。

#### 5个。最后模型与训练

下面是模型的代码以及预训练嵌入。我排除了训练循环代码。



#### torchtext中的其他类

Torchtext还提供类，用于加载其他类型的数据，如语言建模、序列标记、翻译等。

This wraps up the short discussion on torchtext for sentiment analysis task. It was an overview of torchtext. In the next post I will discuss about implementing Attention for sentence classification.

#### References

[1] <https://arxiv.org/abs/1801.06146>

[2] <http://thinknook.com/wp-content/uploads/2012/09/Sentiment-Analysis-Dataset.zip>

[3] <http://anie.me/On-Torchtext/>