In [1]:

```
# Copyright 2019 Google Inc.

# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at

#     http://www.apache.org/licenses/LICENSE-2.0

# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
```

Predicting Movie Review Sentiment with BERT on TF Hub

使用BERT与TF Hub进行影评情感分析

情感分析

分析文本中所表达的情绪，一般判断是积极的，还是消极的 **是一个分类问题。**

建立文本分类器的方法

- 将文本表示为计算机可以理解的方式
  - 建立一个索引表，把每个词都用一个数字表示
  - 使用一个Dict来存储词汇，本质也是文本数字化
- 以词汇索引序列为输入，以类别为输出建立分类器
  - 本文使用神经网络建立分类器

文本的数字化表征

离散表示

```
- One-hot表示： One-hot简称读热向量编码，也是特征工程中最常用的方法。
- 词袋模型: 词袋模型(Bag-of-words model)，像是句子或是文件这样的文字可以用一个袋子装着这
些词的方式表现，这种表现方式不考虑文法以及词的顺序。
- TF-IDF（term frequency–inverse document frequency）是一种用于信息检索与数据挖掘的常用
加权技术。TF意思是词频(Term Frequency)，IDF意思是逆文本频率指数(Inverse Document Frequen
cy)。
- n-gram模型： n-gram模型为了保持词的顺序，做了一个滑窗的操作，这里的n表示的就是滑窗的大
小，例如2-gram模型，也就是把2个词当做一组来处理，然后向后移动一个词的长度，再次组成另一
组词，把这些生成一个字典，按照词袋模型的方式进行编码得到结果。改模型考虑了词的顺序。
```

One-hot表示

- 构造文本分词后的字典，每个分词是一个比特值，比特值为0或者1。
- 每个分词的文本表示为该分词的比特位为1，其余位为0的矩阵表示。

例如：John likes to watch movies. Mary likes too John also likes to watch football games. 以上两句可以构造一个词典，**{"John": 1, "likes": 2, "to": 3, "watch": 4, "movies": 5, "also": 6, "football": 7, "games": 8, "Mary": 9, "too": 10}** 每个词典索引对应着比特位。那么利用One-hot表示为： **John: [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]** likes: [0, 1, 0, 0, 0, 0, 0, 0, 0, 0] ....等等，以此类推。 One-hot表示文本信息的缺点：

随着语料库的增加，数据特征的维度会越来越大，产生一个维度很高，又很稀疏的矩阵。 这种表示方法的分词顺序和在句子中的顺序是无关的，不能保留词与词之间的关系信息。

词袋模型

文档的向量表示可以直接将各词的词向量表示加和。例如： John likes to watch movies. Mary likes too John also likes to watch football games. 以上两句可以构造一个词典，**{"John": 1, "likes": 2, "to": 3, "watch": 4, "movies": 5, "also": 6, "football": 7, "games": 8, "Mary": 9, "too": 10}** 那么第一句的向量表示为：[1,2,1,1,1,0,0,0,1,1]，其中的2表示likes在该句中出现了2次，依次类推。 词袋模型同样有一下缺点：

- 词向量化后，词与词之间是有大小关系的，不一定词出现的越多，权重越大。
- 词与词之间是没有顺序关系的。

TF-IDF

字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在整个语料库中出现的频率成反比下降。一个词语在一篇文章中出现次数越多, 同时在所有文档中出现次数越少, 越能够代表该文章。

一些词，如“*的、地、得*”，出现很多，但不重要。而另外的词，如“*工业革命*”就比较重要。

$$TF_w = \frac{\text{在某一类中词条}w\text{出现的次数}}{\text{该类中所有的词条数目}}$$

$$IDF = log(\frac{语料库的文档总数}{包含词条w的文档总数+1})$$

分母之所以加1，是为了避免分母为0。 那么，，从这个公式可以看出，**当w在文档中出现的次数增大时，而TF-IDF的值是减小的，所以也就体现了以上所说的了。**

缺点：**还是没有把词与词之间的关系顺序表达出来。**

n-gram模型

考虑了顺序
John likes to watch movies. Mary likes too John also likes to watch football games. 以上两句可以构造一个词典，{"John likes": 1, "likes to": 2, "to watch": 3, "watch movies": 4, "Mary likes": 5, "likes too": 6, "John also": 7, "also likes": 8, “watch football”: 9, "football games": 10} 那么第一句的向量表示为：[1, 1, 1, 1, 1, 1, 0, 0, 0, 0]，其中第一个1表示John likes在该句中出现了1次，依次类推。
缺点：
随着n的大小增加，词表会成指数型膨胀，会越来越大。

离散表示存在的问题

由于存在以下的问题，对于一般的NLP问题，是可以使用离散表示文本信息来解决问题的，但对于要求精度较高的场景就不适合了。

无法衡量词向量之间的关系。 词表的维度随着语料库的增长而膨胀。 n-gram词序列随语料库增长呈指数型膨胀，更加快。 离散数据来表示文本会带来数据稀疏问题，导致丢失了信息，与我们生活中理解的信息是不一样

神经网络表示

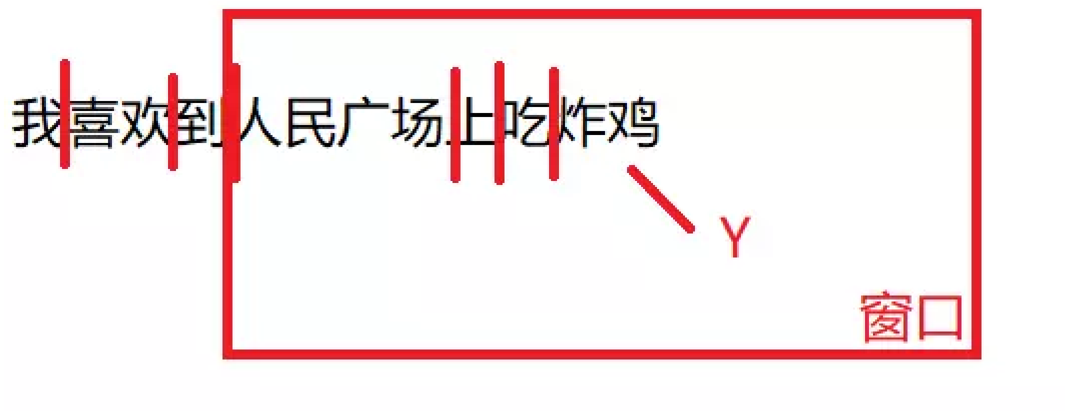
NNLM

NNLM (Neural Network Language model)，神经网络语言模型是03年提出来的，通过训练得到中间产物--词向量矩阵，这就是我们要得到的文本表示向量矩阵。 NNLM说的是定义一个前向窗口大小，其实和上面提到的窗口是一个意思。把这个窗口中最后一个词当做y，把之前的词当做输入x，通俗来说就是预测这个窗口中最后一个词出现概率的模型。

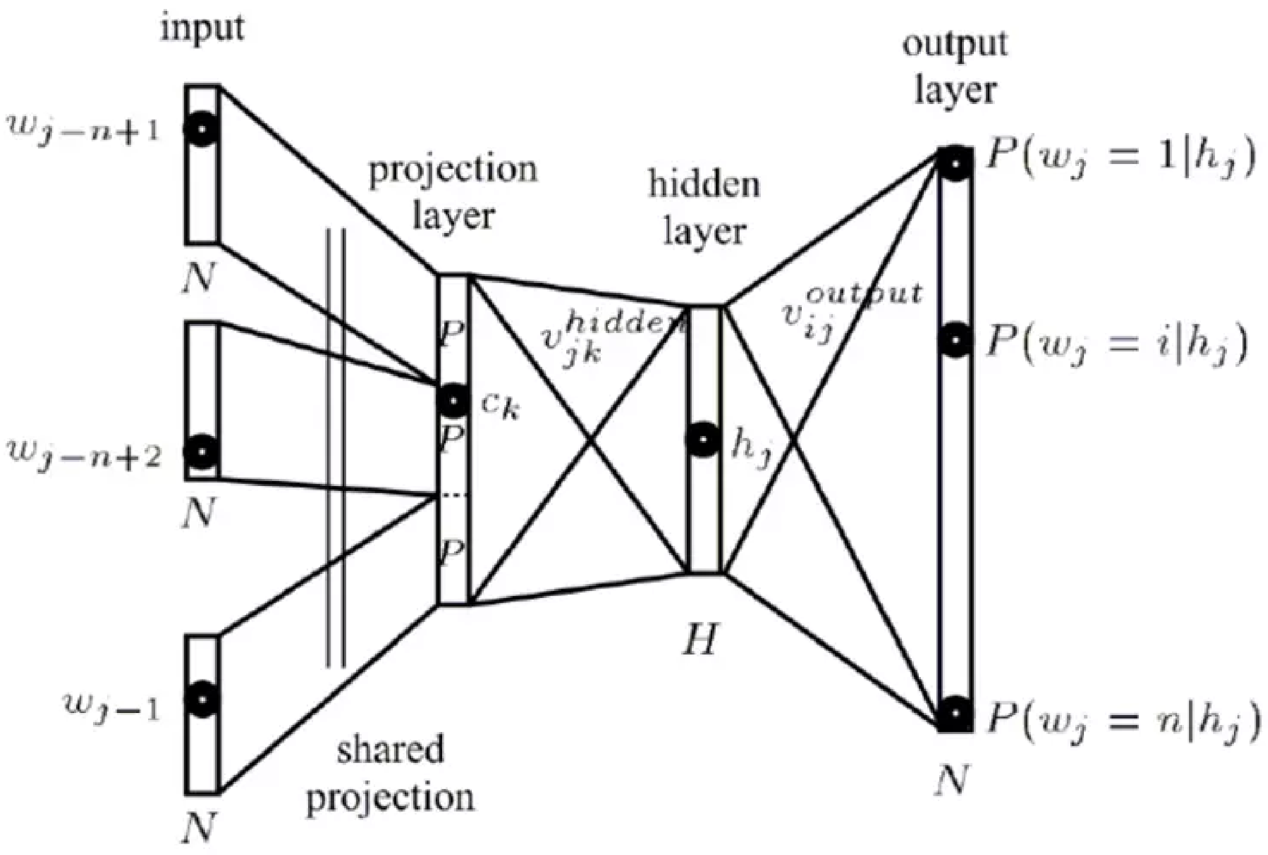
语言模型是这样一个模型：对于任意的词序列，它能够计算出这个序列是一句话的概率。语言模型是假设一门语言所有可能的句子服从一个概率分布，每个句子出现的概率加起来是1，那么语言模型的任务就是预测每个句子在语言中出现的概率，对于语言中常见的句子，一个好的语言模型应该得到相对高的概率，对不合语法的句子，计算出的概率则趋近于零。如果把句子$(w_1, w_2, \ldots, w_m)$看成单词的序列，那么语言模型可以表示为一个计算$p\left(w_1, w_2, \ldots, w_m\right)$的模型。语言模型仅仅对句子出现的概率进行建模，并不尝试去理解句子的内容含义。

- 经典神经网络模型的输入输出

将前N-1个词对应的词向量首尾拼接作为神经网络的输入，神经网络的输出有V个节点，这些节点的输出经过softmax激活函数后，就归一化为给定前N-1个词，下一个词是该词的条件概率，于是就得到了N-Gram模型的参数，也就构建了N-Gram模型。
模型结构包括两部分，分别是Embedding层和前馈神经网络。
Embedding层：负责将词语转化为词向量。(我们所需要的)
前馈神经网络：这是一个三层的前馈神经网络，包括输入层，隐藏层和输出层，都是全连接层。输入层有(N-1)m个节点，隐藏层有h个节点，输出层有V个节点，隐藏层的激活函数是双曲正切函数tanh()，这是一个S型函数，输出层的激活函数是softmax。
大部分机器学习算法其实都是一个最优化问题，神经网络也不例外。这里的目标函数是语料库的对数似然函数加上一个正则项，优化变量是模型参数，通过随机梯度提升的方法，调整参数，使得目标函数达到最大即可。目标函数中的语料库对数似然函数其实就是对数条件概率的累加，也就是模型的输出取对数后的累加。



以下是NNLM的网络结构图：



input层是一个前向词的输入，是经过one-hot编码的词向量表示形式，具有V*1的矩阵。

C矩阵是投影矩阵，也就是稠密词向量表示，在神经网络中是w参数矩阵，该矩阵的大小为DV，*正好与input层进行全连接(相乘)得到D*1的矩阵，采用线性映射将one-hot表 示投影到稠密D维表示。

C矩阵是投影矩阵，也是稠密词向量表示

$$C=(w_1,w_2,\cdots,w_V)=\begin{pmatrix} (w_1)_1 & \boxed{(w_2)_1} & \cdots & (w_V)_1 \\ (w_1)_2 & (w_1)_2 & \cdots & (w_V)_2 \\ \vdots & \vdots & \ddots & \vdots \\ (w_1)_D & (w_1)_D & \cdots & (w_V)_D \end{pmatrix}$$

output层(softmax)自然是前向窗中需要预测的词。

通过BP＋SGD得到最优的C投影矩阵，这就是NNLM的中间产物，也是我们所求的文本表示矩阵，通过NNLM将稀疏矩阵投影到稠密向量矩阵中。

Word2Vec

谷歌2013年提出的Word2Vec是目前最常用的词嵌入模型之一。Word2Vec实际 是一种浅层的神经网络模型，它有两种网络结构，**分别是CBOW（Continues Bag of Words）连续词袋和Skip-gram。**Word2Vec和上面的NNLM很类似，但比NNLM简单。

- CBOW CBOW获得**中间词两边的的上下文，然后用周围的词去预测中间的词**，*把中间词当做y，把窗口中的其它词当做x输入*，x输入是经过one-hot编码过的，然后通过一个隐层进行求和操作，最后通过激活函数softmax，可以计算出每个单词的生成概率，接下来的任务就是训练神经网络的权重，使得语料库中所有单词的整体生成概率最大化，而**求得的权重矩阵就是文本表示词向量的结果。**
- Skip-gram：Skip-gram是通过当前词来预测窗口中上下文词出现的概率模型，把当前词当做x，把窗口中其它词当做y，依然是通过一个隐层接一个Softmax激活函数来预测其它词的概率。

Word2Vec存在的问题

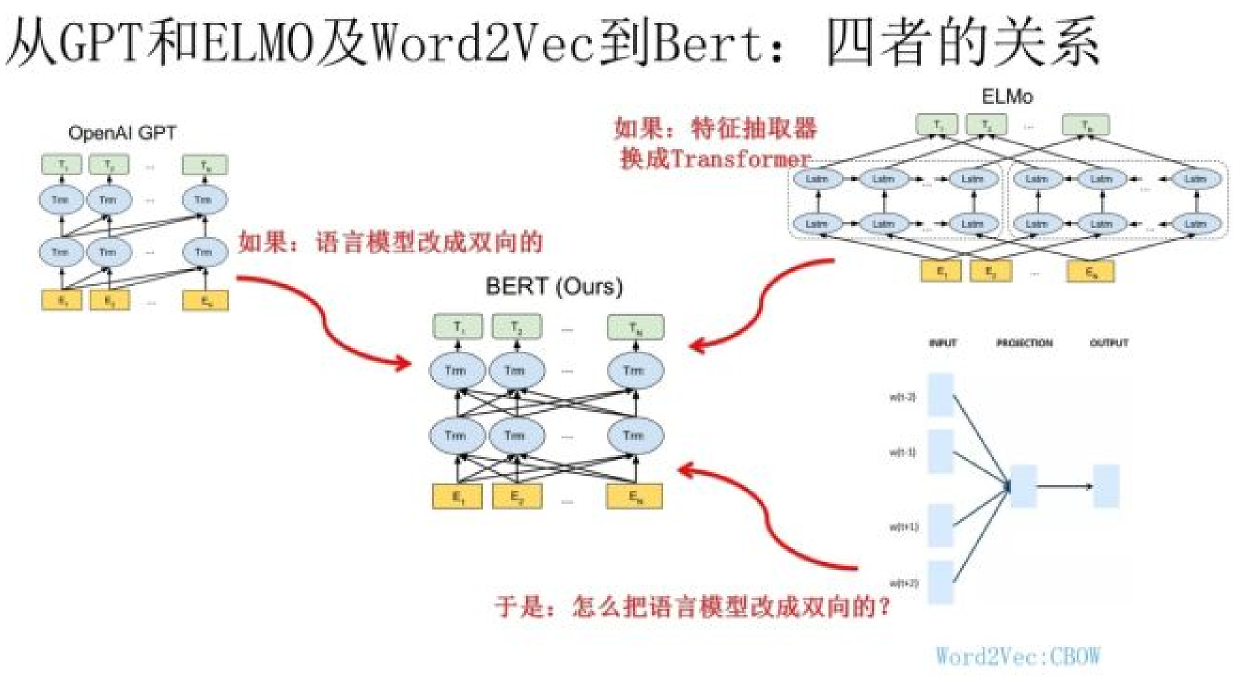
对每个local context window单独训练，没有利用包 含在global co-currence矩阵中的统计信息。 对多义词无法很好的表示和处理，因为使用了唯一 的词向量

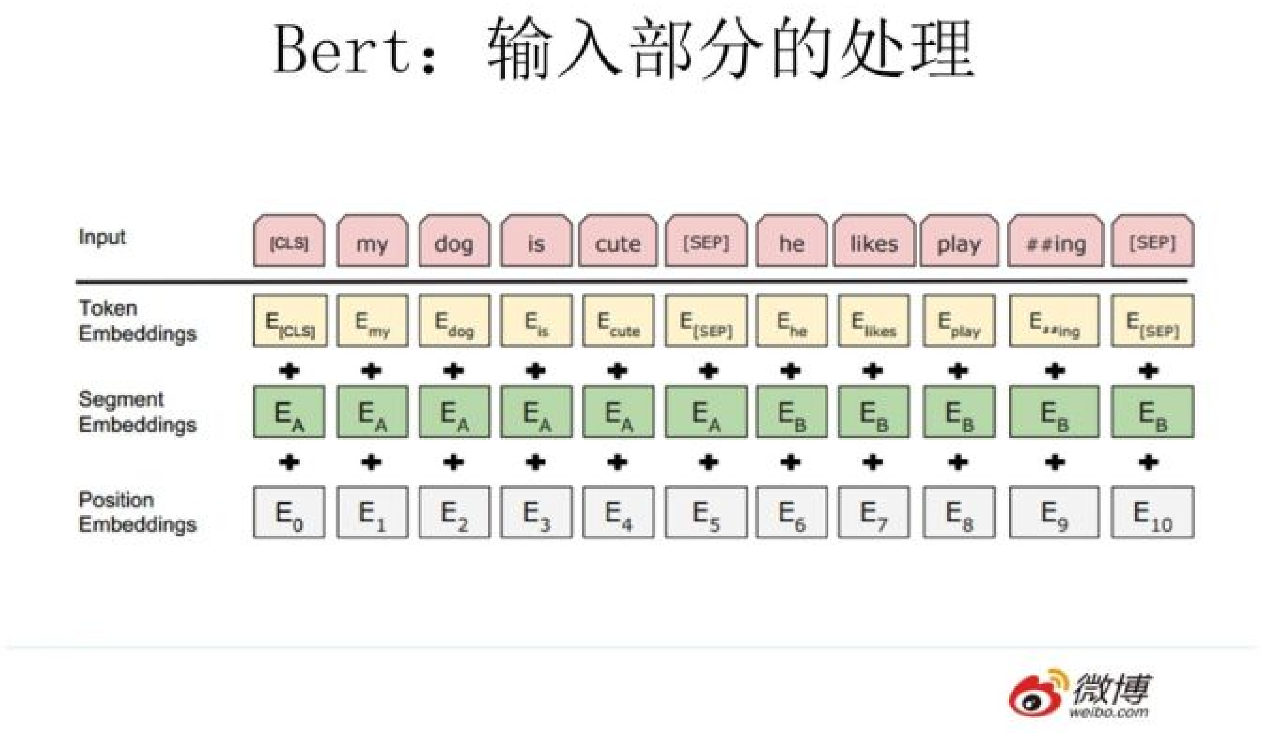
BERT

双向语言模型

NNLM 是单向的，意思是只能根据上文推下文，而BERT可以根据后文推前文，类似完形填空。

至于完形填空要猜哪个词，由mask技术控制。 Masked双向语言模型向上图展示这么做：随机选择语料中15%的单词，把它抠掉，也就是用[Mask]掩码代替原始单词，然后要求模型去正确预测被抠掉的单词。但是这里有个问题：训练过程大量看到[mask]标记，但是真正后面用的时候是不会有这个标记的，这会引导模型认为输出是针对[mask]这个标记的，但是实际使用又见不到这个标记，这自然会有问题。为了避免这个问题，Bert改造了一下，15%的被上天选中要执行[mask]替身这项光荣任务的单词中，只有80%真正被替换成[mask]标记，10%被狸猫换太子随机替换成另外一个单词，10%情况这个单词还待在原地不做改动。这就是Masked双向语音模型的具体做法。


从GPT和ELMO及Word2Vec到Bert：四者的关系
OpenAI GPT
ELMo
如果：特征抽取器换成Transformer
如果：语言模型改成双向的
BERT (Ours)
于是：怎么把语言模型改成双向的？
INPUT
PROJECTION
OUTPUT
Word2Vec:CBOW



注意力机制

顾名思义，注意力机制是本质上是为了模仿人类观察物品的方式。通常来说，人们在看一张图片的时候，除了从整体把握一幅图片之外，也会更加关注图片的某个局部信息，例如局部桌子的位置，商品的种类等等。在翻译领域，每次人们翻译一段话的时候，通常都是从句子入手，但是在阅读整个句子的时候，肯定就需要关注词语本身的信息，以及词语前后关系的信息和上下文的信息。在自然语言处理方向，如果要进行情感分类的话，在某个句子里面，肯定会涉及到表达情感的词语，包括但不限于“高兴”，“沮丧”，“开心”等关键词。而这些句子里面的其他词语，则是上下文的关系，并不是它们没有用，而是它们所起的作用没有那些表达情感的关键词大。

在以上描述下，注意力机制其实包含两个部分：

注意力机制需要决定整段输入的哪个部分需要更加关注；

从关键的部分进行特征提取，得到重要的信息。

通常来说，在机器翻译或者自然语言处理领域，人们阅读和理解一句话或者一段话其实是有着一定的先后顺序的，并且按照语言学的语法规则来进行阅读理解。在图片分类领域，人们看一幅图也是按照先整体再局部，或者先局部再整体来看的。再看局部的时候，尤其是手写的手机号，门牌号等信息，都是有先后顺序的。为了模拟人脑的思维方式和理解模式，循环神经网络（RNN）在处理这种具有明显先后顺序的问题上有着独特的优势，因此，Attention 机制通常都会应用在循环神经网络上面。

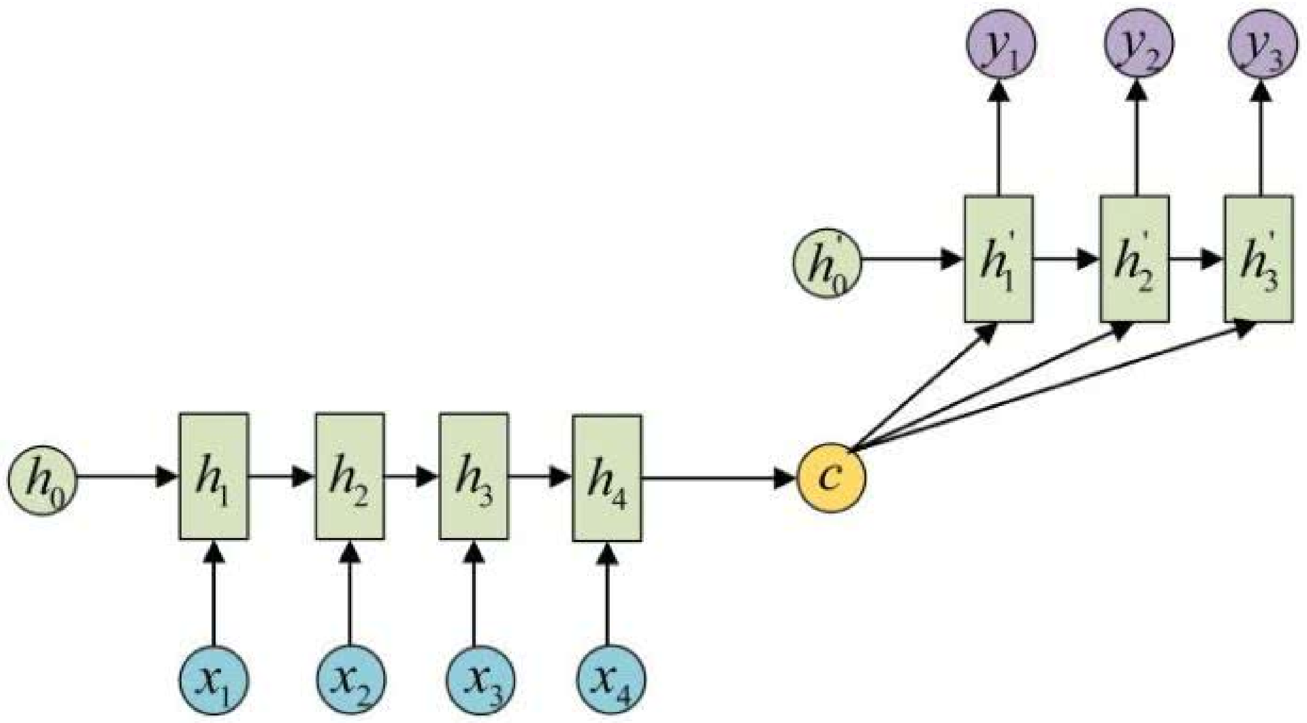
虽然，按照上面的描述，机器翻译，自然语言处理，计算机视觉领域的注意力机制差不多，但是其实仔细推敲起来，这三者的注意力机制是有明显区别的。

在机器翻译领域，翻译人员需要把已有的一句话翻译成另外一种语言的一句话。例如把一句话从英文翻译到中文，把中文翻译到法语。在这种情况下，输入语言和输出语言的词语之间的先后顺序其实是相对固定的，是具有一定的语法规则的；

在视频分类或者情感识别领域，视频的先后顺序是由时间戳和相应的片段组成的，输入的就是一段视频里面的关键片段，也就是一系列具有先后顺序的图片的组合。NLP 中的情感识别问题也是一样的，语言本身就具有先后顺序的特点；

图像识别，物体检测领域与前面两个有本质的不同。因为物体检测其实是在一幅图里面挖掘出必要的物体结构或者位置信息，在这种情况下，它的输入就是一幅图片，并没有非常明显的先后顺序，而且从人脑的角度来看，由于个体的差异性，很难找到一个通用的观察图片的方法。由于每个人都有着自己观察的先后顺序，因此很难统一成一个整体。

在这种情况下，机器翻译和自然语言处理领域使用基于 RNN 的 Attention 机制就变得相对自然，而计算机视觉领域领域则需要必要的改造才能够使用 Attention 机制。



Transform

BERT创新点在于抛弃了之前传统的encoder-decoder模型必须结合cnn或者rnn的固有模式，只用attention。提出了两个新的Attention机制，分别叫做 Scaled Dot-Product Attention 和 Multi-Head Attention。提出了transformer，是BERT的一大特点。

总结一下，我们需要获得文本的数字化表征方式，而BERT是目前最优秀的一个，所以我们选BERT来将文本数字化，再在其基础上构建分类器等进行后续任务。

代码实战

TensorFlow Hub

TensorFlow Hub 是一个针对可重复使用的机器学习模块的库。

TensorFlow Hub 是一个库，用于发布、发现和使用机器学习模型中可重复利用的部分。模块是**一个独立的 TensorFlow 图部分，其中包含权重和资源**，*可以在一个进程中供不同任务重复使用（称为迁移学习）。*

迁移学习可以:

- 使用较小的数据集训练模型，
- 改善泛化效果，以及 -加快训练速度。

神经网络架构，可以用于分类、翻译、总结和问答。

BERT现在已经加入了TF Hub， 可以更容易的加入现有的Tensorflow文本流水线，用于取代如ELMO和GloVE的文本映射表征层。如果加上微调，BERT可以更高的准确度和训练速度。

如果本地计算资源不足，可以在colab上用云服务器运行。在线colab notebook (https://colab.sandbox.google.com/github/tensorflow/tpu/blob/master/tools/colab/bert_finetuning_with_cloud_tp

我们在这里会用TF Hub上的BERT训练一个IMDB的影评模型，预测消极或积极情绪。

If you’ve been following Natural Language Processing over the past year, you’ve probably heard of **BERT: Bidirectional Encoder Representations from Transformers**. It’s a neural network architecture designed by Google researchers that’s totally transformed what’s state-of-the-art for NLP tasks, like text classification, translation, summarization, and question answering.

Now that BERT's been added to TF Hub (https://www.tensorflow.org/hub) as a loadable module, it's easy(ish) to add into existing Tensorflow text pipelines. In an existing pipeline, BERT can replace text embedding layers like ELMO and GloVE. Alternatively, finetuning (http://wiki.fast.ai/index.php/Fine_tuning) BERT can provide both an accuracy boost and faster training time in many cases.

Here, we'll train a model to predict whether an IMDB movie review is positive or negative using BERT in Tensorflow with tf hub. Some code was adapted from this colab notebook (https://colab.sandbox.google.com/github/tensorflow/tpu/blob/master/tools/colab/bert_finetuning_with_cloud_tp
Let's get started!

In [10]:

```
#安装所需要的库
!pip install tensorflow_hub
!pip install "tf-nightly"
!pip install sklearn
!pip install pandas
```

```
Collecting tensorflow_hub
  Using cached https://files.pythonhosted.org/packages/00/0e/a91780d07592b1abf9c91
344ce459472cc19db3b67fdf3a61dca6ebb2f5c/tensorflow_hub-0.7.0-py2.py3-none-any.whl
Requirement already satisfied: numpy>=1.12.0 in f:\programming\anaconda3\envs\bert
\lib\site-packages (from tensorflow_hub) (1.17.4)
Requirement already satisfied: six>=1.10.0 in f:\programming\anaconda3\envs\bert\l
ib\site-packages (from tensorflow_hub) (1.13.0)
Requirement already satisfied: protobuf>=3.4.0 in f:\programming\anaconda3\envs\be
rt\lib\site-packages (from tensorflow_hub) (3.10.0)
Requirement already satisfied: setuptools in f:\programming\anaconda3\envs\bert\li
b\site-packages (from protobuf>=3.4.0->tensorflow_hub) (41.6.0.post20191030)
Installing collected packages: tensorflow-hub
Successfully installed tensorflow-hub-0.7.0
Collecting tf-nightly
  Downloading https://files.pythonhosted.org/packages/ae/6f/55fab6908ada59d23aa288
7556e4b0a5d83f5a6d2a0fc3af51c2e1cbf22c/tf_nightly-2.1.0.dev20191116-cp36-cp36m-win
_amd64.whl (64.6MB)
Requirement already satisfied: opt-einsum>=2.3.2 in f:\programming\anaconda3\envs
\bert\lib\site-packages (from tf-nightly) (3.1.0)
Requirement already satisfied: astor>=0.6.0 in f:\programming\anaconda3\envs\bert
\lib\site-packages (from tf-nightly) (0.8.0)
Requirement already satisfied: keras-applications>=1.0.8 in f:\programming\anacond
a3\envs\bert\lib\site-packages (from tf-nightly) (1.0.8)
Requirement already satisfied: six>=1.12.0 in f:\programming\anaconda3\envs\bert\l
ib\site-packages (from tf-nightly) (1.13.0)
Requirement already satisfied: protobuf>=3.8.0 in f:\programming\anaconda3\envs\be
rt\lib\site-packages (from tf-nightly) (3.10.0)
Requirement already satisfied: grpcio>=1.8.6 in f:\programming\anaconda3\envs\bert
\lib\site-packages (from tf-nightly) (1.25.0)
Collecting tb-nightly<2.2.0a0,>=2.1.0a0
  Using cached https://files.pythonhosted.org/packages/c9/b5/0503fd8e8656f7cfd765c
7b8d52760a78478643b3134870e40657a139f60/tb_nightly-2.1.0a20191116-py3-none-any.whl
Collecting tf-estimator-nightly
  Using cached https://files.pythonhosted.org/packages/32/78/0409cbf5e11315109886f
890e4f03cc283f1d22e7e91cf8edb18c8261626/tf_estimator_nightly-2.0.0.dev2019111609-p
y2.py3-none-any.whl
Requirement already satisfied: termcolor>=1.1.0 in f:\programming\anaconda3\envs\b
ert\lib\site-packages (from tf-nightly) (1.1.0)
Requirement already satisfied: keras-preprocessing>=1.1.0 in f:\programming\anacon
da3\envs\bert\lib\site-packages (from tf-nightly) (1.1.0)
Requirement already satisfied: absl-py>=0.7.0 in f:\programming\anaconda3\envs\ber
t\lib\site-packages (from tf-nightly) (0.8.1)
Requirement already satisfied: numpy<2.0,>=1.16.0 in f:\programming\anaconda3\envs
\bert\lib\site-packages (from tf-nightly) (1.17.4)
Requirement already satisfied: google-pasta>=0.1.8 in f:\programming\anaconda3\env
s\bert\lib\site-packages (from tf-nightly) (0.1.8)
Requirement already satisfied: gast==0.2.2 in f:\programming\anaconda3\envs\bert\l
ib\site-packages (from tf-nightly) (0.2.2)
Requirement already satisfied: wrapt>=1.11.1 in f:\programming\anaconda3\envs\bert
\lib\site-packages (from tf-nightly) (1.11.2)
Requirement already satisfied: wheel>=0.26; python_version >= "3" in f:\programmin
g\anaconda3\envs\bert\lib\site-packages (from tf-nightly) (0.33.6)
Requirement already satisfied: h5py in f:\programming\anaconda3\envs\bert\lib\site
-packages (from keras-applications>=1.0.8->tf-nightly) (2.10.0)
Requirement already satisfied: setuptools in f:\programming\anaconda3\envs\bert\li
b\site-packages (from protobuf>=3.8.0->tf-nightly) (41.6.0.post20191030)
Requirement already satisfied: requests<3,>=2.21.0 in f:\programming\anaconda3\env
s\bert\lib\site-packages (from tb-nightly<2.2.0a0,>=2.1.0a0->tf-nightly) (2.22.0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in f:\programming
\anaconda3\envs\bert\lib\site-packages (from tb-nightly<2.2.0a0,>=2.1.0a0->tf-nigh
tly) (0.4.1)
```

```
Requirement already satisfied: google-auth<2,>=1.6.3 in f:\programming\anaconda3\e
nvs\bert\lib\site-packages (from tb-nightly<2.2.0a0,>=2.1.0a0->tf-nightly) (1.7.1)
Requirement already satisfied: werkzeug>=0.11.15 in f:\programming\anaconda3\envs
\bert\lib\site-packages (from tb-nightly<2.2.0a0,>=2.1.0a0->tf-nightly) (0.16.0)
Requirement already satisfied: markdown>=2.6.8 in f:\programming\anaconda3\envs\be
rt\lib\site-packages (from tb-nightly<2.2.0a0,>=2.1.0a0->tf-nightly) (3.1.1)
Requirement already satisfied: certifi>=2017.4.17 in f:\programming\anaconda3\envs
\bert\lib\site-packages (from requests<3,>=2.21.0->tb-nightly<2.2.0a0,>=2.1.0a0->t
f-nightly) (2019.9.11)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in f:\programming\anaconda3\e
nvs\bert\lib\site-packages (from requests<3,>=2.21.0->tb-nightly<2.2.0a0,>=2.1.0a0
->tf-nightly) (3.0.4)
Requirement already satisfied: idna<2.9,>=2.5 in f:\programming\anaconda3\envs\ber
t\lib\site-packages (from requests<3,>=2.21.0->tb-nightly<2.2.0a0,>=2.1.0a0->tf-ni
ghtly) (2.8)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in f:\progr
amming\anaconda3\envs\bert\lib\site-packages (from requests<3,>=2.21.0->tb-nightly
<2.2.0a0,>=2.1.0a0->tf-nightly) (1.25.7)
Requirement already satisfied: requests-oauthlib>=0.7.0 in f:\programming\anaconda
3\envs\bert\lib\site-packages (from google-auth-oauthlib<0.5,>=0.4.1->tb-nightly<
2.2.0a0,>=2.1.0a0->tf-nightly) (1.3.0)
Requirement already satisfied: pyasn1-modules>=0.2.1 in f:\programming\anaconda3\e
nvs\bert\lib\site-packages (from google-auth<2,>=1.6.3->tb-nightly<2.2.0a0,>=2.1.0
a0->tf-nightly) (0.2.7)
Requirement already satisfied: cachetools<3.2,>=2.0.0 in f:\programming\anaconda3
\envs\bert\lib\site-packages (from google-auth<2,>=1.6.3->tb-nightly<2.2.0a0,>=2.
1.0a0->tf-nightly) (3.1.1)
Requirement already satisfied: rsa<4.1,>=3.1.4 in f:\programming\anaconda3\envs\be
rt\lib\site-packages (from google-auth<2,>=1.6.3->tb-nightly<2.2.0a0,>=2.1.0a0->tf
-nightly) (4.0)
Requirement already satisfied: oauthlib>=3.0.0 in f:\programming\anaconda3\envs\be
rt\lib\site-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.
4.1->tb-nightly<2.2.0a0,>=2.1.0a0->tf-nightly) (3.1.0)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in f:\programming\anaconda3\en
vs\bert\lib\site-packages (from pyasn1-modules>=0.2.1->google-auth<2,>=1.6.3->tb-n
ightly<2.2.0a0,>=2.1.0a0->tf-nightly) (0.4.8)
Installing collected packages: tb-nightly, tf-estimator-nightly, tf-nightly
Successfully installed tb-nightly-2.1.0a20191116 tf-estimator-nightly-2.0.0.dev201
9111609 tf-nightly-2.1.0.dev20191116
```

In [1]:

```
#安装tensorflow gpu 版 tf最好比1.14新，以保证对tf hub的支持
!pip install tensorflow-gpu==1.15
```

```
Collecting tensorflow-gpu==1.15
  Using cached https://files.pythonhosted.org/packages/93/2e/72862c0a79d11d0b3cd1f
c512fadab2c268454ac2d477ca2ab11926e2122/tensorflow_gpu-1.15.0-cp36-cp36m-win_amd6
4.whl
Requirement already satisfied: wrapt>=1.11.1 in f:\programming\anaconda3\envs\bert
\lib\site-packages (from tensorflow-gpu==1.15) (1.11.2)
Requirement already satisfied: protobuf>=3.6.1 in f:\programming\anaconda3\envs\be
rt\lib\site-packages (from tensorflow-gpu==1.15) (3.10.0)
Requirement already satisfied: tensorflow-estimator==1.15.1 in f:\programming\anac
onda3\envs\bert\lib\site-packages (from tensorflow-gpu==1.15) (1.15.1)
Requirement already satisfied: absl-py>=0.7.0 in f:\programming\anaconda3\envs\ber
t\lib\site-packages (from tensorflow-gpu==1.15) (0.8.1)
Requirement already satisfied: keras-applications>=1.0.8 in f:\programming\anacond
a3\envs\bert\lib\site-packages (from tensorflow-gpu==1.15) (1.0.8)
Requirement already satisfied: grpcio>=1.8.6 in f:\programming\anaconda3\envs\bert
\lib\site-packages (from tensorflow-gpu==1.15) (1.25.0)
Requirement already satisfied: gast==0.2.2 in f:\programming\anaconda3\envs\bert\l
ib\site-packages (from tensorflow-gpu==1.15) (0.2.2)
Requirement already satisfied: keras-preprocessing>=1.0.5 in f:\programming\anacon
da3\envs\bert\lib\site-packages (from tensorflow-gpu==1.15) (1.1.0)
Requirement already satisfied: numpy<2.0,>=1.16.0 in f:\programming\anaconda3\envs
\bert\lib\site-packages (from tensorflow-gpu==1.15) (1.17.4)
Requirement already satisfied: tensorboard<1.16.0,>=1.15.0 in f:\programming\anaco
nda3\envs\bert\lib\site-packages (from tensorflow-gpu==1.15) (1.15.0)
Requirement already satisfied: opt-einsum>=2.3.2 in f:\programming\anaconda3\envs
\bert\lib\site-packages (from tensorflow-gpu==1.15) (3.1.0)
Requirement already satisfied: google-pasta>=0.1.6 in f:\programming\anaconda3\env
s\bert\lib\site-packages (from tensorflow-gpu==1.15) (0.1.8)
Requirement already satisfied: six>=1.10.0 in f:\programming\anaconda3\envs\bert\l
ib\site-packages (from tensorflow-gpu==1.15) (1.13.0)
Requirement already satisfied: wheel>=0.26 in f:\programming\anaconda3\envs\bert\l
ib\site-packages (from tensorflow-gpu==1.15) (0.33.6)
Requirement already satisfied: termcolor>=1.1.0 in f:\programming\anaconda3\envs\b
ert\lib\site-packages (from tensorflow-gpu==1.15) (1.1.0)
Requirement already satisfied: astor>=0.6.0 in f:\programming\anaconda3\envs\bert
\lib\site-packages (from tensorflow-gpu==1.15) (0.8.0)
Requirement already satisfied: setuptools in f:\programming\anaconda3\envs\bert\li
b\site-packages (from protobuf>=3.6.1->tensorflow-gpu==1.15) (41.6.0.post20191030)
Requirement already satisfied: h5py in f:\programming\anaconda3\envs\bert\lib\site
-packages (from keras-applications>=1.0.8->tensorflow-gpu==1.15) (2.10.0)
Requirement already satisfied: werkzeug>=0.11.15 in f:\programming\anaconda3\envs
\bert\lib\site-packages (from tensorboard<1.16.0,>=1.15.0->tensorflow-gpu==1.15)
(0.16.0)
Requirement already satisfied: markdown>=2.6.8 in f:\programming\anaconda3\envs\be
rt\lib\site-packages (from tensorboard<1.16.0,>=1.15.0->tensorflow-gpu==1.15) (3.
1.1)
Installing collected packages: tensorflow-gpu
  Found existing installation: tensorflow-gpu 2.0.0
    Uninstalling tensorflow-gpu-2.0.0:
      Successfully uninstalled tensorflow-gpu-2.0.0
Successfully installed tensorflow-gpu-1.15.0
```

In [2]:

```
from sklearn.model_selection import train_test_split
import pandas as pd
import tensorflow as tf
import tensorflow_hub as hub
from datetime import datetime
```

In addition to the standard libraries we imported above, we'll need to install BERT's python package.

In [12]:

```
!pip install bert-tensorflow
```

```
Collecting bert-tensorflow
  Using cached https://files.pythonhosted.org/packages/a6/66/7eb4e8b6ea35b7cc54c32
2c816f976167a43019750279a8473d355800a93/bert_tensorflow-1.0.1-py2.py3-none-any.whl
Requirement already satisfied: six in f:\programming\anaconda3\envs\bert\lib\site-
packages (from bert-tensorflow) (1.13.0)
Installing collected packages: bert-tensorflow
Successfully installed bert-tensorflow-1.0.1
```

In [3]:

```
import bert
from bert import run_classifier
from bert import optimization
from bert import tokenization
```

```
WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\bert
\optimization.py:87: The name tf.train.Optimizer is deprecated. Please use tf.comp
at.v1.train.Optimizer instead.
```

设置模型保存位置

Tensorflow中，训练好的模型可以输出并保存到本地，以便下次直接使用。

Google Cloud Platform云平台用户可以选择GCP bucket。

Below, we'll set an output directory location to store our model output and checkpoints. This can be a local directory, in which case you'd set OUTPUT_DIR to the name of the directory you'd like to create. If you're running this code in Google's hosted Colab, the directory won't persist after the Colab session ends.

Alternatively, if you're a GCP user, you can store output in a GCP bucket. To do that, set a directory name in OUTPUT_DIR and the name of the GCP bucket in the BUCKET field.

Set DO_DELETE to rewrite the OUTPUT_DIR if it exists. Otherwise, Tensorflow will load existing model checkpoints from that directory (if they exist).

In [14]:

```
# Set the output directory for saving model file 模型保存位置
# Optionally, set a GCP bucket location

OUTPUT_DIR = 'OUTPUT_Bert'#@param {type:"string"}
#@markdown Whether or not to clear/delete the directory and create a new one
DO_DELETE = False #@param {type:"boolean"}
#@markdown Set USE_BUCKET and BUCKET if you want to (optionally) store model output on GCP bucke
t. 如果要输出GCP bucket
USE_BUCKET = False #@param {type:"boolean"}
BUCKET = 'BUCKET_NAME' #@param {type:"string"}

if USE_BUCKET:
  OUTPUT_DIR = 'gs://{}/{}'.format(BUCKET, OUTPUT_DIR)
  from google.colab import auth
  auth.authenticate_user()

if DO_DELETE:
  try:
    tf.gfile.DeleteRecursively(OUTPUT_DIR)
  except:
    # Doesn't matter if the directory didn't exist
    pass
tf.gfile.MakeDirs(OUTPUT_DIR)
print('***** Model output directory: {} *****'.format(OUTPUT_DIR))
```

```
***** Model output directory: OUTPUT_Bert *****
```

Data

First, let's download the dataset, hosted by Stanford. The code below, which downloads, extracts, and imports the IMDB Large Movie Review Dataset, is borrowed from this Tensorflow tutorial (https://www.tensorflow.org/hub/tutorials/text_classification_with_tf_hub).

数据导入

In [4]:

```
from tensorflow import keras #基于tf的高层库keras
import os
import re

# Load all files from a directory in a DataFrame.
def load_directory_data(directory):
  data = {}
  data["sentence"] = []
  data["sentiment"] = []
  for file_path in os.listdir(directory):
    with tf.gfile.GFile(os.path.join(directory, file_path), "r") as f:
      data["sentence"].append(f.read())
      data["sentiment"].append(re.match("\d+_(\d+)\.txt", file_path).group(1))
  return pd.DataFrame.from_dict(data)

# Merge positive and negative examples, add a polarity column and shuffle.
def load_dataset(directory):
  pos_df = load_directory_data(os.path.join(directory, "pos"))
  neg_df = load_directory_data(os.path.join(directory, "neg"))
  pos_df["polarity"] = 1
  neg_df["polarity"] = 0
  return pd.concat([pos_df, neg_df]).sample(frac=1).reset_index(drop=True)

# Download and process the dataset files.
def download_and_load_datasets(force_download=False):
  dataset = tf.keras.utils.get_file(
      fname="aclImdb.tar.gz",
      origin="http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz",
      extract=True)#数据地址，下载解压

  train_df = load_dataset(os.path.join(os.path.dirname(dataset),
                                       "aclImdb", "train")) #训练集，在train文件夹下
  test_df = load_dataset(os.path.join(os.path.dirname(dataset),
                                      "aclImdb", "test"))

  return train_df, test_df
```

In [5]:

```
#train, test = download_and_load_datasets()
```

In [36]:

```
#如果网络不好，从本地加载数据
train = load_dataset('aclImdb_v1\\train')
test  = load_dataset('aclImdb_v1\\test')
```

To keep training fast, we'll take a sample of 5000 train and test examples, respectively.

为了训练快些，只各取5000训练数据和测试数据

In [17]:

```
train = train.sample(5000)
test = test.sample(5000)
```

In [40]:

```
train.columns
```

Out[40]:

```
Index(['sentence', 'sentiment', 'polarity'], dtype='object')
```

For us, our input data is the 'sentence' column and our label is the 'polarity' column (0, 1 for negative and positive, respecitvely)

文本在sentence中，label是polarity（0表示消极，1积极）

In [18]:

```
DATA_COLUMN = 'sentence'
LABEL_COLUMN = 'polarity'
# label_list is the list of labels, i.e. True, False or 0, 1 or 'dog', 'cat'
label_list = [0, 1]
```

Data Preprocessing

We'll need to transform our data into a format BERT understands. This involves two steps. First, we create `InputExample`'s using the constructor provided in the BERT library.

- `text_a` is the text we want to classify, which in this case, is the `Request` field in our Dataframe.
- `text_b` is used if we're training a model to understand the relationship between sentences (i.e. is `text_b` a translation of `text_a`? Is `text_b` an answer to the question asked by `text_a`?). This doesn't apply to our task, so we can leave `text_b` blank.
- `label` is the label for our example, i.e. True, False

数据预处理

我们需要讲文本转化为BERT可以理解的方式。

- `text_a` 需要分类的文本。
- `text_b` 需要知道和`text_a` 关系的文本，例如在问答任务中，本任务不需要了解。
- `label` is the 标签 for our example, i.e. True, False

In [19]:

```
# Use the InputExample class from BERT's run_classifier code to create examples from the data
#准备好训练、测试数据
train_InputExamples = train.apply(lambda x: bert.run_classifier.InputExample(guid=None, # Globa
lly unique ID for bookkeeping, unused in this example
                                                                   text_a = x[DATA_COLUMN],
                                                                   text_b = None,
                                                                   label = x[LABEL_COLUMN]), axi
s = 1)

test_InputExamples = test.apply(lambda x: bert.run_classifier.InputExample(guid=None,
                                                                   text_a = x[DATA_COLUMN],
                                                                   text_b = None,
                                                                   label = x[LABEL_COLUMN]), axi
s = 1)
```

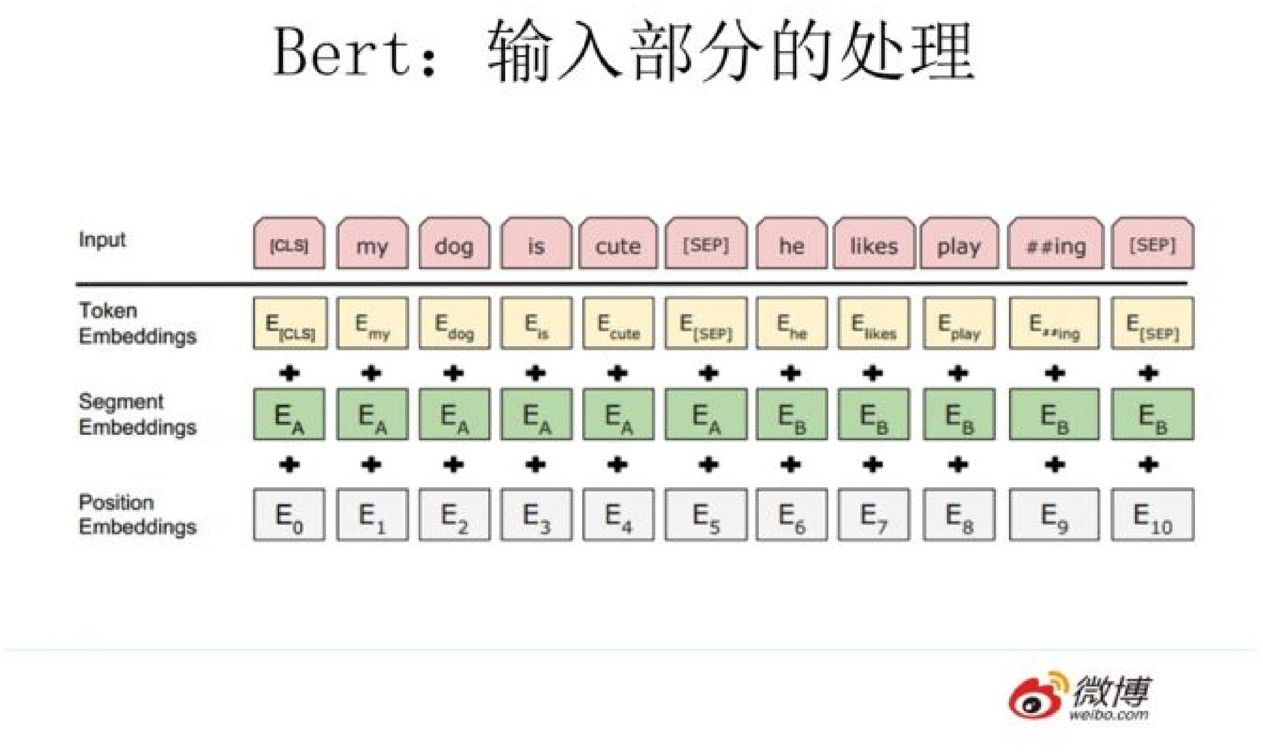
Next, we need to preprocess our data so that it matches the data BERT was trained on. For this, we'll need to do a couple of things (but don't worry--this is also included in the Python library):

数据进一步处理。

1. Lowercase our text (if we're using a BERT lowercase model)
2. Tokenize it (i.e. "sally says hi" -> ["sally", "says", "hi"])
3. Break words into WordPieces (i.e. "calling" -> ["call", "##ing"])
4. Map our words to indexes using a vocab file that BERT provides
5. Add special "CLS" and "SEP" tokens (see the readme (https://github.com/google-research/bert)) CLS表示句子开头，SEP表示句子结尾。
6. Append "index" and "segment" tokens to each input (see the BERT paper (https://arxiv.org/pdf/1810.04805.pdf)) 标注句子index

Happily, we don't have to worry about most of these details.

1. 文本统一为小写
2. Tokenize处理 (i.e. "sally says hi" -> ["sally", "says", "hi"])
3. Break words into WordPieces 词语细粒度切分 (i.e. "calling" -> ["call", "##ing"])
4. 将词汇单元映射为数字索引，保证神经网络输入为数字形式。
5. 添加句子开头结尾标志 (see the readme (https://github.com/google-research/bert)) CLS表示句子开头，SEP表示句子结尾。
6. (see the BERT paper (https://arxiv.org/pdf/1810.04805.pdf)) 标注每个句子的序号。


Bert：输入部分的处理
Input
[CLS] my dog is cute [SEP] he likes play ##ing [SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}
微博 weibo.com

To start, we'll need to load a vocabulary file and lowercasing information directly from the BERT tf hub module:

需要从tf hub下载一个词汇表文件来帮助索引

In [20]:

```
# This is a path to an uncased (all lowercase) version of BERT
#BERT_MODEL_HUB = "https://tfhub.dev/google/bert_uncased_L-12_H-768_A-12/1"
BERT_MODEL_HUB = 'BERT_MODEL_HUB'
def create_tokenizer_from_hub_module():
  """Get the vocab file and casing info from the Hub module."""
  with tf.Graph().as_default():
    bert_module = hub.Module(BERT_MODEL_HUB)
    #module_spec =  hub.load_module_spec(BERT_MODEL_HUB) #从本地载入
    #bert_module = hub.Module(module_spec,trainable=True)#这个很关键,后面有个trainable=True,如果是True这个网络继续训练,如果是False,这个网络就不训练了,只训练你后添加的哪些网络层
    tokenization_info = bert_module(signature="tokenization_info", as_dict=True)
    with tf.Session() as sess:
      vocab_file, do_lower_case = sess.run([tokenization_info["vocab_file"],
                                            tokenization_info["do_lower_case"]])

  return bert.tokenization.FullTokenizer(
      vocab_file=vocab_file, do_lower_case=do_lower_case)

tokenizer = create_tokenizer_from_hub_module()
```

```
INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore
```

```
INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore
```

Great--we just learned that the BERT model we're using expects lowercase data (that's what stored in tokenization_info["do_lower_case"]) and we also loaded BERT's vocab file. We also created a tokenizer, which breaks words into word pieces:

将词分割为词片段

In [22]:

```
tokenizer.tokenize("This here's an example of using the BERT tokenizer")
```

Out[22]:

```
['this',
 'here',
 "'",
 's',
 'an',
 'example',
 'of',
 'using',
 'the',
 'bert',
 'token',
 '##izer']
```

Using our tokenizer, we'll call `run_classifier.convert_examples_to_features` on our InputExamples to convert them into features BERT understands.

将所有数据转化BERT定义的输入。

In [65]:

```
print(train.iloc[0])
print(train['sentence'].iloc[0])
print(train_features[0].input_ids) #输入的第一个序列对应的数字索引
print(train_features[0].input_mask)#input_mask 是input中需要mask的位置，本来是随机取一部分，这里的做法是把全部输入位置都mask住。
#这个mask 和之前说的单词的mask 意义是不太一样的，这个地方的mask是为了统一batch_size中不同的instance 的长度的，不够长的就补0，然后使用input_mask 来记录instance的真实长度
#1 代表是真实的单词id ，0代表补全位
print(train_features[0].is_real_example)
print(train_features[0].label_id)
print(train_features[0].segment_ids) #segment_ids存储的是句子的id ，id 为0 就是第一句，id 为1 就是第二句
```

```
sentence     This movie will tell you why Amitabh Bacchan i...
sentiment                                                  8
polarity                                                   1
Name: 8738, dtype: object
This movie will tell you why Amitabh Bacchan is a one man industry. This movie wil
l also tell you why Indian movie-goers are astute buyers.<br /><br />Amitabh was a
t the peak of his domination of Bollywood when his one-time godfather Prakash Mehr
a decided to use his image yet again. Prakash has the habit of picking themes and
building stories out of it, adding liberal doses of Bollywood sensibilities and cl
ichés to it. Zanzeer saw the making of Angry Young Man. Lawaris was about being a
bastard and Namak Halal was about the master-servant loyalties. <br /><br />But th
en, the theme was limited to move the screenplay through the regulation three hour
s of song, dance and drama. What comprised of the movie is a caricature of a Harya
navi who goes to Mumbai and turns into a regulation hero. Amitabh's vocal skills a
nd diction saw this movie earn its big bucks, thanks to his flawless stock Haryanv
i accent. To me, this alone is the biggest pull in the movie. The rest all is typi
cal Bollywood screen writing.<br /><br />Amitabh, by now, had to have some typical
comedy scenes in each of his movies. Thanks to Manmohan Desai. This movie had a go
od dose of them. The shoe caper in the party, the monologue over Vijay Merchant an
d Vijay Hazare's considerations, The mosquito challenge in the boardroom and the u
sual drunkard scene that by now has become a standard Amitabh fare.<br /><br />Sha
shi Kapoor added an extra mile to the movie with his moody, finicky character (Rem
ember him asking Ranjeet to "Shaaadaaaap" after the poisoned cake incident"). His
was the all important role of the master while Amitabh was his loyal servant. But
Prakash Mehra knew the Indian mind...and so Shashi had to carry along his act with
the rest of the movie. It was one character that could have been more developed to
make a serious movie. But this is a caper, remember? And as long as it stayed that
way, the people came and saw Amitabh wearing a new hat and went back home happy. T
he end is always predictable, and the good guys get the gal and the bad ones go to
the gaol, the age-old theme of loyalty is once again emphasized and all is well th
at ends well.<br /><br />So what is it that makes this movie a near classic? Amita
bh Bacchan as the Haryanvi. Prakash Mehra created yet another icon in the name of
a story. Chuck the story, the characters and the plot. My marks are for Amitabh al
one.
[101, 2023, 3185, 2097, 2425, 2017, 2339, 26445, 2696, 23706, 8670, 9468, 4819, 20
03, 1037, 2028, 2158, 3068, 1012, 2023, 3185, 2097, 2036, 2425, 2017, 2339, 2796,
3185, 1011, 2175, 2545, 2024, 2004, 24518, 17394, 1012, 1026, 7987, 1013, 1028, 10
26, 7987, 1013, 1028, 26445, 2696, 23706, 2001, 2012, 1996, 4672, 1997, 2010, 1788
2, 1997, 16046, 2043, 2010, 2028, 1011, 2051, 23834, 22233, 2033, 13492, 2787, 200
0, 2224, 2010, 3746, 2664, 2153, 1012, 22233, 2038, 1996, 10427, 1997, 8130, 6991,
1998, 2311, 3441, 2041, 1997, 2009, 1010, 5815, 4314, 21656, 1997, 16046, 12411, 5
332, 14680, 1998, 18856, 17322, 2015, 2000, 2009, 1012, 23564, 14191, 11510, 2387,
1996, 2437, 1997, 4854, 2402, 2158, 1012, 2375, 23061, 2001, 2055, 2108, 1037, 844
4, 1998, 15125, 4817, 11085, 2389, 2001, 2055, 102]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
True
1
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

In [29]:

```
# We'll set sequences to be at most 128 tokens long. 序列最长为128
MAX_SEQ_LENGTH = 128
# Convert our train and test features to InputFeatures that BERT understands.
train_features = bert.run_classifier.convert_examples_to_features(train_InputExamples, label_lis
t, MAX_SEQ_LENGTH, tokenizer)
test_features = bert.run_classifier.convert_examples_to_features(test_InputExamples, label_list,
MAX_SEQ_LENGTH, tokenizer)
```

```
INFO:tensorflow:Writing example 0 of 5000
```

```
INFO:tensorflow:Writing example 0 of 5000
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] i am a big fan of arnold vo ##sl ##oo . finally seei
ng him as the star of a recent movie , not just a bit part , made me happy . < br
/ > < br / > unfortunately i took film appreciation in college and the only thing
i can say that i didn ' t like was that the film was made in an abandoned part of
town and there was no background traffic or look ##ie lo ##os . < br / > < br / >
i have to say that the acting leaves something to be desired , but arnold is an ex
cellent actor , i have to chalk it up to lou ##sy direction and the supporting cas
t [SEP]
```

```
INFO:tensorflow:tokens: [CLS] i am a big fan of arnold vo ##sl ##oo . finally seei
ng him as the star of a recent movie , not just a bit part , made me happy . < br
/ > < br / > unfortunately i took film appreciation in college and the only thing
i can say that i didn ' t like was that the film was made in an abandoned part of
town and there was no background traffic or look ##ie lo ##os . < br / > < br / >
i have to say that the acting leaves something to be desired , but arnold is an ex
cellent actor , i have to chalk it up to lou ##sy direction and the supporting cas
t [SEP]
```

```
INFO:tensorflow:input_ids: 101 1045 2572 1037 2502 5470 1997 7779 29536 14540 9541
1012 2633 3773 2032 2004 1996 2732 1997 1037 3522 3185 1010 2025 2074 1037 2978 21
12 1010 2081 2033 3407 1012 1026 7987 1013 1028 1026 7987 1013 1028 6854 1045 2165
2143 12284 1999 2267 1998 1996 2069 2518 1045 2064 2360 2008 1045 2134 1005 1056 2
066 2001 2008 1996 2143 2001 2081 1999 2019 4704 2112 1997 2237 1998 2045 2001 205
3 4281 4026 2030 2298 2666 8840 2891 1012 1026 7987 1013 1028 1026 7987 1013 1028
1045 2031 2000 2360 2008 1996 3772 3727 2242 2000 2022 9059 1010 2021 7779 2003 20
19 6581 3364 1010 1045 2031 2000 16833 2009 2039 2000 10223 6508 3257 1998 1996 46
37 3459 102
```

```
INFO:tensorflow:input_ids: 101 1045 2572 1037 2502 5470 1997 7779 29536 14540 9541
1012 2633 3773 2032 2004 1996 2732 1997 1037 3522 3185 1010 2025 2074 1037 2978 21
12 1010 2081 2033 3407 1012 1026 7987 1013 1028 1026 7987 1013 1028 6854 1045 2165
2143 12284 1999 2267 1998 1996 2069 2518 1045 2064 2360 2008 1045 2134 1005 1056 2
066 2001 2008 1996 2143 2001 2081 1999 2019 4704 2112 1997 2237 1998 2045 2001 205
3 4281 4026 2030 2298 2666 8840 2891 1012 1026 7987 1013 1028 1026 7987 1013 1028
1045 2031 2000 2360 2008 1996 3772 3727 2242 2000 2022 9059 1010 2021 7779 2003 20
19 6581 3364 1010 1045 2031 2000 16833 2009 2039 2000 10223 6508 3257 1998 1996 46
37 3459 102
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] this is what i call a great movie . it lives trough
the fantastic actor skills and a simple but human story . there are real character
s which can be funny and dramatic . but the main theme is very cruel , like live i
s . the bus driver and his son are collecting people trough the country ( jug ##os
lav ##ia ) on their way to the capital bel ##grad . the funny and cruel situations
that happens on the way , connect the people and the pigs that travel together . <
br / > < br / > watch it and you gonna remember it for life . . . its filled with
slavic humor and lifestyle . < br / > < [SEP]
```

```
INFO:tensorflow:tokens: [CLS] this is what i call a great movie . it lives trough
the fantastic actor skills and a simple but human story . there are real character
s which can be funny and dramatic . but the main theme is very cruel , like live i
s . the bus driver and his son are collecting people trough the country ( jug ##os
lav ##ia ) on their way to the capital bel ##grad . the funny and cruel situations
that happens on the way , connect the people and the pigs that travel together . <
br / > < br / > watch it and you gonna remember it for life . . . its filled with
slavic humor and lifestyle . < br / > < [SEP]
```

```
INFO:tensorflow:input_ids: 101 2023 2003 2054 1045 2655 1037 2307 3185 1012 2009 3
268 23389 1996 10392 3364 4813 1998 1037 3722 2021 2529 2466 1012 2045 2024 2613 3
494 2029 2064 2022 6057 1998 6918 1012 2021 1996 2364 4323 2003 2200 10311 1010 20
66 2444 2003 1012 1996 3902 4062 1998 2010 2365 2024 9334 2111 23389 1996 2406 100
6 26536 23085 2401 1007 2006 2037 2126 2000 1996 3007 19337 16307 1012 1996 6057 1
998 10311 8146 2008 6433 2006 1996 2126 1010 7532 1996 2111 1998 1996 14695 2008 3
604 2362 1012 1026 7987 1013 1028 1026 7987 1013 1028 3422 2009 1998 2017 6069 334
2 2009 2005 2166 1012 1012 1012 2049 3561 2007 13838 8562 1998 9580 1012 1026 7987
1013 1028 1026 102
```

```
INFO:tensorflow:input_ids: 101 2023 2003 2054 1045 2655 1037 2307 3185 1012 2009 3
268 23389 1996 10392 3364 4813 1998 1037 3722 2021 2529 2466 1012 2045 2024 2613 3
494 2029 2064 2022 6057 1998 6918 1012 2021 1996 2364 4323 2003 2200 10311 1010 20
66 2444 2003 1012 1996 3902 4062 1998 2010 2365 2024 9334 2111 23389 1996 2406 100
6 26536 23085 2401 1007 2006 2037 2126 2000 1996 3007 19337 16307 1012 1996 6057 1
998 10311 8146 2008 6433 2006 1996 2126 1010 7532 1996 2111 1998 1996 14695 2008 3
604 2362 1012 1026 7987 1013 1028 1026 7987 1013 1028 3422 2009 1998 2017 6069 334
2 2009 2005 2166 1012 1012 1012 2049 3561 2007 13838 8562 1998 9580 1012 1026 7987
1013 1028 1026 102
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 1 (id = 1)
```

```
INFO:tensorflow:label: 1 (id = 1)
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] what can you possibly say about a show of this magni
tude ? " the soprano ##s " has literally red ##efined television as we know it . i
t has broken all rules , and set new standards for television excellence . everyth
ing is flawless , the writing , directing , and for me , most of all , the acting
. watching this show you ' ll find yourself realizing that these characters are no
t real . the acting tricks you into thinking there is a real tony soprano , or any
character . this show is also very versatile . some people don ' t watch the show
because it ' s violent , it ' s not all about the violence , it ' [SEP]
```

```
INFO:tensorflow:tokens: [CLS] what can you possibly say about a show of this magni
tude ? " the soprano ##s " has literally red ##efined television as we know it . i
t has broken all rules , and set new standards for television excellence . everyth
ing is flawless , the writing , directing , and for me , most of all , the acting
. watching this show you ' ll find yourself realizing that these characters are no
t real . the acting tricks you into thinking there is a real tony soprano , or any
character . this show is also very versatile . some people don ' t watch the show
because it ' s violent , it ' s not all about the violence , it ' [SEP]
```

```
INFO:tensorflow:input_ids: 101 2054 2064 2017 4298 2360 2055 1037 2265 1997 2023 1
0194 1029 1000 1996 10430 2015 1000 2038 6719 2417 28344 2547 2004 2057 2113 2009
1012 2009 2038 3714 2035 3513 1010 1998 2275 2047 4781 2005 2547 8012 1012 2673 20
03 27503 1010 1996 3015 1010 9855 1010 1998 2005 2033 1010 2087 1997 2035 1010 199
6 3772 1012 3666 2023 2265 2017 1005 2222 2424 4426 9301 2008 2122 3494 2024 2025
2613 1012 1996 3772 12225 2017 2046 3241 2045 2003 1037 2613 4116 10430 1010 2030
2151 2839 1012 2023 2265 2003 2036 2200 22979 1012 2070 2111 2123 1005 1056 3422 1
996 2265 2138 2009 1005 1055 6355 1010 2009 1005 1055 2025 2035 2055 1996 4808 101
0 2009 1005 102
```

```
INFO:tensorflow:input_ids: 101 2054 2064 2017 4298 2360 2055 1037 2265 1997 2023 1
0194 1029 1000 1996 10430 2015 1000 2038 6719 2417 28344 2547 2004 2057 2113 2009
1012 2009 2038 3714 2035 3513 1010 1998 2275 2047 4781 2005 2547 8012 1012 2673 20
03 27503 1010 1996 3015 1010 9855 1010 1998 2005 2033 1010 2087 1997 2035 1010 199
6 3772 1012 3666 2023 2265 2017 1005 2222 2424 4426 9301 2008 2122 3494 2024 2025
2613 1012 1996 3772 12225 2017 2046 3241 2045 2003 1037 2613 4116 10430 1010 2030
2151 2839 1012 2023 2265 2003 2036 2200 22979 1012 2070 2111 2123 1005 1056 3422 1
996 2265 2138 2009 1005 1055 6355 1010 2009 1005 1055 2025 2035 2055 1996 4808 101
0 2009 1005 102

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 1 (id = 1)
```

```
INFO:tensorflow:label: 1 (id = 1)
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] when i go to see movies i would stay up and watch it
or if i did not like it , i would go sleep , but this was pure crap , i actually g
ot up and walked out ! . . . . this was poorly script and put together , i hated i
t . also , they should not have taken brendan fra ##sier off , he was much better
. this was not as good as i had expected , considering that i really liked george
of the jungle 1 , and the graphics weren ’ t as good as the first one , for instan
ce , the bird , and when ever he crashed in a tree . i hope that the [SEP]
```

```
INFO:tensorflow:tokens: [CLS] when i go to see movies i would stay up and watch it
or if i did not like it , i would go sleep , but this was pure crap , i actually g
ot up and walked out ! . . . . this was poorly script and put together , i hated i
t . also , they should not have taken brendan fra ##sier off , he was much better
. this was not as good as i had expected , considering that i really liked george
of the jungle 1 , and the graphics weren ’ t as good as the first one , for instan
ce , the bird , and when ever he crashed in a tree . i hope that the [SEP]
```

```
INFO:tensorflow:input_ids: 101 2043 1045 2175 2000 2156 5691 1045 2052 2994 2039 1
998 3422 2009 2030 2065 1045 2106 2025 2066 2009 1010 1045 2052 2175 3637 1010 202
1 2023 2001 5760 10231 1010 1045 2941 2288 2039 1998 2939 2041 999 1012 1012 1012
1012 2023 2001 9996 5896 1998 2404 2362 1010 1045 6283 2009 1012 2036 1010 2027 23
23 2025 2031 2579 15039 25312 20236 2125 1010 2002 2001 2172 2488 1012 2023 2001 2
025 2004 2204 2004 1045 2018 3517 1010 6195 2008 1045 2428 4669 2577 1997 1996 889
4 1015 1010 1998 1996 8389 4694 1005 1056 2004 2204 2004 1996 2034 2028 1010 2005
6013 1010 1996 4743 1010 1998 2043 2412 2002 8007 1999 1037 3392 1012 1045 3246 20
08 1996 102
```

```
INFO:tensorflow:input_ids: 101 2043 1045 2175 2000 2156 5691 1045 2052 2994 2039 1
998 3422 2009 2030 2065 1045 2106 2025 2066 2009 1010 1045 2052 2175 3637 1010 202
1 2023 2001 5760 10231 1010 1045 2941 2288 2039 1998 2939 2041 999 1012 1012 1012
1012 2023 2001 9996 5896 1998 2404 2362 1010 1045 6283 2009 1012 2036 1010 2027 23
23 2025 2031 2579 15039 25312 20236 2125 1010 2002 2001 2172 2488 1012 2023 2001 2
025 2004 2204 2004 1045 2018 3517 1010 6195 2008 1045 2428 4669 2577 1997 1996 889
4 1015 1010 1998 1996 8389 4694 1005 1056 2004 2204 2004 1996 2034 2028 1010 2005
6013 1010 1996 4743 1010 1998 2043 2412 2002 8007 1999 1037 3392 1012 1045 3246 20
08 1996 102
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] it ’ s difficult to precisely put into words the she
er awful ##ness of this film . an entirely new vocabulary will have to be invented
to describe the complete absence of anything even remotely recognizable as ’ humor
’ or even ’ entertainment ’ in ″ rabbit test . ″ so , as a small contribution to t
his future effort , i ’ d like to suggest this word : < br / > < br / > ″ hub ##ir
i ##ffi ##c ″ ( ad ##j . ) a combination of ’ hub ##rist ##ic ’ and ’ terrific ’ ;
used to describe overly ambitious de ##ba ##cles like the film ″ rabbit test . ″ <
br / > < br / [SEP]
```

```
INFO:tensorflow:tokens: [CLS] it ' s difficult to precisely put into words the she
er awful ##ness of this film . an entirely new vocabulary will have to be invented
to describe the complete absence of anything even remotely recognizable as ' humor
' or even ' entertainment ' in " rabbit test . " so , as a small contribution to t
his future effort , i ' d like to suggest this word : < br / > < br / > " hub ##ir
i ##ffi ##c " ( ad ##j . ) a combination of ' hub ##rist ##ic ' and ' terrific ' ;
used to describe overly ambitious de ##ba ##cles like the film " rabbit test . " <
br / > < br / [SEP]

INFO:tensorflow:input_ids: 101 2009 1005 1055 3697 2000 10785 2404 2046 2616 1996
11591 9643 2791 1997 2023 2143 1012 2019 4498 2047 16188 2097 2031 2000 2022 8826
2000 6235 1996 3143 6438 1997 2505 2130 19512 20123 2004 1005 8562 1005 2030 2130
1005 4024 1005 1999 1000 10442 3231 1012 1000 2061 1010 2004 1037 2235 6691 2000 2
023 2925 3947 1010 1045 1005 1040 2066 2000 6592 2023 2773 1024 1026 7987 1013 102
8 1026 7987 1013 1028 1000 9594 15735 26989 2278 1000 1006 4748 3501 1012 1007 103
7 5257 1997 1005 9594 15061 2594 1005 1998 1005 27547 1005 1025 2109 2000 6235 152
41 12479 2139 3676 18954 2066 1996 2143 1000 10442 3231 1012 1000 1026 7987 1013 1
028 1026 7987 1013 102
```

```
INFO:tensorflow:input_ids: 101 2009 1005 1055 3697 2000 10785 2404 2046 2616 1996
11591 9643 2791 1997 2023 2143 1012 2019 4498 2047 16188 2097 2031 2000 2022 8826
2000 6235 1996 3143 6438 1997 2505 2130 19512 20123 2004 1005 8562 1005 2030 2130
1005 4024 1005 1999 1000 10442 3231 1012 1000 2061 1010 2004 1037 2235 6691 2000 2
023 2925 3947 1010 1045 1005 1040 2066 2000 6592 2023 2773 1024 1026 7987 1013 102
8 1026 7987 1013 1028 1000 9594 15735 26989 2278 1000 1006 4748 3501 1012 1007 103
7 5257 1997 1005 9594 15061 2594 1005 1998 1005 27547 1005 1025 2109 2000 6235 152
41 12479 2139 3676 18954 2066 1996 2143 1000 10442 3231 1012 1000 1026 7987 1013 1
028 1026 7987 1013 102
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:Writing example 0 of 5000

INFO:tensorflow:Writing example 0 of 5000


INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid: None

INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] if i could give this movie a negative rating i would
. the humor is the cruel ##est i have ever seen in a film . horrible things happen
to good people and people who have already suffered horribly through no fault of t
heir own . there are 2 plots , neither of which supports half of a film . where is
the " depth " others see in this movie ? that no good deed goes un ##pu ##nished ?
that only the heart ##less can succeed ? the film does start well and the black an
d white is very moody and well done . the acting is very good and convincing witch
makes the cruel humor even more ho ##rri ##fying . if you [SEP]
```

```
INFO:tensorflow:tokens: [CLS] if i could give this movie a negative rating i would
. the humor is the cruel ##est i have ever seen in a film . horrible things happen
to good people and people who have already suffered horribly through no fault of t
heir own . there are 2 plots , neither of which supports half of a film . where is
the " depth " others see in this movie ? that no good deed goes un ##pu ##nished ?
that only the heart ##less can succeed ? the film does start well and the black an
d white is very moody and well done . the acting is very good and convincing witch
makes the cruel humor even more ho ##rri ##fying . if you [SEP]
```

```
INFO:tensorflow:input_ids: 101 2065 1045 2071 2507 2023 3185 1037 4997 5790 1045 2
052 1012 1996 8562 2003 1996 10311 4355 1045 2031 2412 2464 1999 1037 2143 1012 92
02 2477 4148 2000 2204 2111 1998 2111 2040 2031 2525 4265 27762 2083 2053 6346 199
7 2037 2219 1012 2045 2024 1016 14811 1010 4445 1997 2029 6753 2431 1997 1037 2143
1012 2073 2003 1996 1000 5995 1000 2500 2156 1999 2023 3185 1029 2008 2053 2204 15
046 3632 4895 14289 28357 1029 2008 2069 1996 2540 3238 2064 9510 1029 1996 2143 2
515 2707 2092 1998 1996 2304 1998 2317 2003 2200 14434 1998 2092 2589 1012 1996 37
72 2003 2200 2204 1998 13359 6965 3084 1996 10311 8562 2130 2062 7570 18752 14116
1012 2065 2017 102
```

```
INFO:tensorflow:input_ids: 101 2065 1045 2071 2507 2023 3185 1037 4997 5790 1045 2
052 1012 1996 8562 2003 1996 10311 4355 1045 2031 2412 2464 1999 1037 2143 1012 92
02 2477 4148 2000 2204 2111 1998 2111 2040 2031 2525 4265 27762 2083 2053 6346 199
7 2037 2219 1012 2045 2024 1016 14811 1010 4445 1997 2029 6753 2431 1997 1037 2143
1012 2073 2003 1996 1000 5995 1000 2500 2156 1999 2023 3185 1029 2008 2053 2204 15
046 3632 4895 14289 28357 1029 2008 2069 1996 2540 3238 2064 9510 1029 1996 2143 2
515 2707 2092 1998 1996 2304 1998 2317 2003 2200 14434 1998 2092 2589 1012 1996 37
72 2003 2200 2204 1998 13359 6965 3084 1996 10311 8562 2130 2062 7570 18752 14116
1012 2065 2017 102
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***

INFO:tensorflow:guid: None

INFO:tensorflow:guid: None

INFO:tensorflow:tokens: [CLS] in his feature film debut ` yellow , ' chris chan le
e attempts to en ##light ##en hollywood ' s portrayal of asian - americans by depa
rting from the stereotypes typically depicted in mainstream film . however , in so
doing , lee commits a far more he ##ino ##us crime : he ex ##agger ##ates asian -
americans ' own stereotypes of themselves to the point of inc ##red ##uli ##ty . t
he result ? dreadful ##ly one - dimensional characters and an outrageous ##ly shal
low script triggers the cast into a frenzy of over - acting , ultimately resulting
in a film that is physically painful to watch . < br / > < br / > don ' t be dec #
#ei ##ved [SEP]

INFO:tensorflow:tokens: [CLS] in his feature film debut ` yellow , ' chris chan le
e attempts to en ##light ##en hollywood ' s portrayal of asian - americans by depa
rting from the stereotypes typically depicted in mainstream film . however , in so
doing , lee commits a far more he ##ino ##us crime : he ex ##agger ##ates asian -
americans ' own stereotypes of themselves to the point of inc ##red ##uli ##ty . t
he result ? dreadful ##ly one - dimensional characters and an outrageous ##ly shal
low script triggers the cast into a frenzy of over - acting , ultimately resulting
in a film that is physically painful to watch . < br / > < br / > don ' t be dec #
#ei ##ved [SEP]

INFO:tensorflow:input_ids: 101 1999 2010 3444 2143 2834 1036 3756 1010 1005 3782 9
212 3389 4740 2000 4372 7138 2368 5365 1005 1055 13954 1997 4004 1011 4841 2011 15
971 2013 1996 22807 4050 8212 1999 7731 2143 1012 2174 1010 1999 2061 2725 1010 33
89 27791 1037 2521 2062 2002 5740 2271 4126 1024 2002 4654 27609 8520 4004 1011 48
41 1005 2219 22807 1997 3209 2000 1996 2391 1997 4297 5596 15859 3723 1012 1996 27
65 1029 21794 2135 2028 1011 8789 3494 1998 2019 25506 2135 8467 5896 27099 1996 3
459 2046 1037 21517 1997 2058 1011 3772 1010 4821 4525 1999 1037 2143 2008 2003 81
86 9145 2000 3422 1012 1026 7987 1013 1028 1026 7987 1013 1028 2123 1005 1056 2022
11703 7416 7178 102

INFO:tensorflow:input_ids: 101 1999 2010 3444 2143 2834 1036 3756 1010 1005 3782 9
212 3389 4740 2000 4372 7138 2368 5365 1005 1055 13954 1997 4004 1011 4841 2011 15
971 2013 1996 22807 4050 8212 1999 7731 2143 1012 2174 1010 1999 2061 2725 1010 33
89 27791 1037 2521 2062 2002 5740 2271 4126 1024 2002 4654 27609 8520 4004 1011 48
41 1005 2219 22807 1997 3209 2000 1996 2391 1997 4297 5596 15859 3723 1012 1996 27
65 1029 21794 2135 2028 1011 8789 3494 1998 2019 25506 2135 8467 5896 27099 1996 3
459 2046 1037 21517 1997 2058 1011 3772 1010 4821 4525 1999 1037 2143 2008 2003 81
86 9145 2000 3422 1012 1026 7987 1013 1028 1026 7987 1013 1028 2123 1005 1056 2022
11703 7416 7178 102

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1


INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] i honestly found wicked little things to be a very c
ool and fun horror film . my friend had given this to me , and i really saw it as
nothing but a crap ##py low class go ##ry horror film . then after i watched this
i was wrong it was very cool and very good and while ill say it seemed a bit unnec
essary at times , and while it may not be the best horror film ever its still good
. i thought the acting was very good especially from the girl who plays the mother
( she seemed very bel ##ie ##vable and to me very li ##ka ##ble ) . and while it i
s a little cl ##iche ’ e and [SEP]
```

```
INFO:tensorflow:tokens: [CLS] i honestly found wicked little things to be a very c
ool and fun horror film . my friend had given this to me , and i really saw it as
nothing but a crap ##py low class go ##ry horror film . then after i watched this
i was wrong it was very cool and very good and while ill say it seemed a bit unnec
essary at times , and while it may not be the best horror film ever its still good
. i thought the acting was very good especially from the girl who plays the mother
( she seemed very bel ##ie ##vable and to me very li ##ka ##ble ) . and while it i
s a little cl ##iche ’ e and [SEP]
```

```
INFO:tensorflow:input_ids: 101 1045 9826 2179 10433 2210 2477 2000 2022 1037 2200
4658 1998 4569 5469 2143 1012 2026 2767 2018 2445 2023 2000 2033 1010 1998 1045 24
28 2387 2009 2004 2498 2021 1037 10231 7685 2659 2465 2175 2854 5469 2143 1012 205
9 2044 1045 3427 2023 1045 2001 3308 2009 2001 2200 4658 1998 2200 2204 1998 2096
5665 2360 2009 2790 1037 2978 14203 2012 2335 1010 1998 2096 2009 2089 2025 2022 1
996 2190 5469 2143 2412 2049 2145 2204 1012 1045 2245 1996 3772 2001 2200 2204 292
6 2013 1996 2611 2040 3248 1996 2388 1006 2016 2790 2200 19337 2666 12423 1998 200
0 2033 2200 5622 2912 3468 1007 1012 1998 2096 2009 2003 1037 2210 18856 17322 100
5 1041 1998 102
```

```
INFO:tensorflow:input_ids: 101 1045 9826 2179 10433 2210 2477 2000 2022 1037 2200
4658 1998 4569 5469 2143 1012 2026 2767 2018 2445 2023 2000 2033 1010 1998 1045 24
28 2387 2009 2004 2498 2021 1037 10231 7685 2659 2465 2175 2854 5469 2143 1012 205
9 2044 1045 3427 2023 1045 2001 3308 2009 2001 2200 4658 1998 2200 2204 1998 2096
5665 2360 2009 2790 1037 2978 14203 2012 2335 1010 1998 2096 2009 2089 2025 2022 1
996 2190 5469 2143 2412 2049 2145 2204 1012 1045 2245 1996 3772 2001 2200 2204 292
6 2013 1996 2611 2040 3248 1996 2388 1006 2016 2790 2200 19337 2666 12423 1998 200
0 2033 2200 5622 2912 3468 1007 1012 1998 2096 2009 2003 1037 2210 18856 17322 100
5 1041 1998 102
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 1 (id = 1)
```

```
INFO:tensorflow:label: 1 (id = 1)
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] 1st watched 6 / 24 / 2007 - 4 out of 10 ( dir - stef
an ru ##jo ##witz ##ky ) : ok thriller , but a little too predictable . this story
is based in germany , which is also where the movie is made . it is about a young
medical student who gets a shot to go to a premiere school in heidelberg and arriv
es seeing some strange things occurring . someone she met on the train there and s
aved , shows up on the school ' s experimentation table and she ' s suspect ##ing
foul play right away . she does some investigation and the disappearance of her fr
iend leads her to a secret society called aaa ( and no it [SEP]
```

```
INFO:tensorflow:tokens: [CLS] 1st watched 6 / 24 / 2007 - 4 out of 10 ( dir - stef
an ru ##jo ##witz ##ky ) : ok thriller , but a little too predictable . this story
is based in germany , which is also where the movie is made . it is about a young
medical student who gets a shot to go to a premiere school in heidelberg and arriv
es seeing some strange things occurring . someone she met on the train there and s
aved , shows up on the school ' s experimentation table and she ' s suspect ##ing
foul play right away . she does some investigation and the disappearance of her fr
iend leads her to a secret society called aaa ( and no it [SEP]
```

```
INFO:tensorflow:input_ids: 101 3083 3427 1020 1013 2484 1013 2289 1011 1018 2041 1
997 2184 1006 16101 1011 8852 21766 5558 15362 4801 1007 1024 7929 10874 1010 2021
1037 2210 2205 21425 1012 2023 2466 2003 2241 1999 2762 1010 2029 2003 2036 2073 1
996 3185 2003 2081 1012 2009 2003 2055 1037 2402 2966 3076 2040 4152 1037 2915 200
0 2175 2000 1037 6765 2082 1999 16793 1998 8480 3773 2070 4326 2477 10066 1012 261
9 2016 2777 2006 1996 3345 2045 1998 5552 1010 3065 2039 2006 1996 2082 1005 1055
21470 2795 1998 2016 1005 1055 8343 2075 12487 2377 2157 2185 1012 2016 2515 2070
4812 1998 1996 13406 1997 2014 2767 5260 2014 2000 1037 3595 2554 2170 13360 1006
1998 2053 2009 102
```

```
INFO:tensorflow:input_ids: 101 3083 3427 1020 1013 2484 1013 2289 1011 1018 2041 1
997 2184 1006 16101 1011 8852 21766 5558 15362 4801 1007 1024 7929 10874 1010 2021
1037 2210 2205 21425 1012 2023 2466 2003 2241 1999 2762 1010 2029 2003 2036 2073 1
996 3185 2003 2081 1012 2009 2003 2055 1037 2402 2966 3076 2040 4152 1037 2915 200
0 2175 2000 1037 6765 2082 1999 16793 1998 8480 3773 2070 4326 2477 10066 1012 261
9 2016 2777 2006 1996 3345 2045 1998 5552 1010 3065 2039 2006 1996 2082 1005 1055
21470 2795 1998 2016 1005 1055 8343 2075 12487 2377 2157 2185 1012 2016 2515 2070
4812 1998 1996 13406 1997 2014 2767 5260 2014 2000 1037 3595 2554 2170 13360 1006
1998 2053 2009 102
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:*** Example ***
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:guid: None
```

```
INFO:tensorflow:tokens: [CLS] i do not understand at all why this movie received such good grades from critics - - i ' ve seen tens of documentaries ( on tv ) about the wine world which were much much better when ( if ) you watch it , please think of two very annoying aspects of mon ##do ##vino : first , the filming is just awful and terrible and upset ##ting : to me , it looked like the guy behind the camera just received the material and was playing with it : plenty of zoom ##s ( for no purpose other than pushing the button in / out ) for instance - - i almost stopped to watch it because of that ! secondly , the interview [SEP]
```

```
INFO:tensorflow:tokens: [CLS] i do not understand at all why this movie received such good grades from critics - - i ' ve seen tens of documentaries ( on tv ) about the wine world which were much much better when ( if ) you watch it , please think of two very annoying aspects of mon ##do ##vino : first , the filming is just awful and terrible and upset ##ting : to me , it looked like the guy behind the camera just received the material and was playing with it : plenty of zoom ##s ( for no purpose other than pushing the button in / out ) for instance - - i almost stopped to watch it because of that ! secondly , the interview [SEP]
```

```
INFO:tensorflow:input_ids: 101 1045 2079 2025 3305 2012 2035 2339 2023 3185 2363 2107 2204 7022 2013 4401 1011 1011 1045 1005 2310 2464 15295 1997 15693 1006 2006 2694 1007 2055 1996 4511 2088 2029 2020 2172 2172 2488 2043 1006 2065 1007 2017 3422 2009 1010 3531 2228 1997 2048 2200 15703 5919 1997 12256 3527 26531 1024 2034 1010 1996 7467 2003 2074 9643 1998 6659 1998 6314 3436 1024 2000 2033 1010 2009 2246 2066 1996 3124 2369 1996 4950 2074 2363 1996 3430 1998 2001 2652 2007 2009 1024 7564 1997 24095 2015 1006 2005 2053 3800 2060 2084 6183 1996 6462 1999 1013 2041 1007 2005 6013 1011 1011 1045 2471 3030 2000 3422 2009 2138 1997 2008 999 16378 1010 1996 4357 102
```

```
INFO:tensorflow:input_ids: 101 1045 2079 2025 3305 2012 2035 2339 2023 3185 2363 2107 2204 7022 2013 4401 1011 1011 1045 1005 2310 2464 15295 1997 15693 1006 2006 2694 1007 2055 1996 4511 2088 2029 2020 2172 2172 2488 2043 1006 2065 1007 2017 3422 2009 1010 3531 2228 1997 2048 2200 15703 5919 1997 12256 3527 26531 1024 2034 1010 1996 7467 2003 2074 9643 1998 6659 1998 6314 3436 1024 2000 2033 1010 2009 2246 2066 1996 3124 2369 1996 4950 2074 2363 1996 3430 1998 2001 2652 2007 2009 1024 7564 1997 24095 2015 1006 2005 2053 3800 2060 2084 6183 1996 6462 1999 1013 2041 1007 2005 6013 1011 1011 1045 2471 3030 2000 3422 2009 2138 1997 2008 999 16378 1010 1996 4357 102
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:label: 0 (id = 0)
```

```
INFO:tensorflow:label: 0 (id = 0)
```

Creating a model

Now that we've prepared our data, let's focus on building a model. `create_model` does just this below. First, it loads the BERT tf hub module again (this time to extract the computation graph). Next, it creates a single new layer that will be trained to adapt BERT to our sentiment task (i.e. classifying whether a movie review is positive or negative). This strategy of using a mostly trained model is called fine-tuning (http://wiki.fast.ai/index.php/Fine_tuning).

在BERT的基础上再加一层训练分类器

In [30]:

```
def create_model(is_predicting, input_ids, input_mask, segment_ids, labels,
                 num_labels):
  """Creates a classification model."""

  bert_module = hub.Module(
      BERT_MODEL_HUB,
      trainable=True) #定义模型
  bert_inputs = dict(
      input_ids=input_ids,
      input_mask=input_mask,
      segment_ids=segment_ids) #定义输入
  bert_outputs = bert_module(
      inputs=bert_inputs,
      signature="tokens",
      as_dict=True) #定义输出

  # Use "pooled_output" for classification tasks on an entire sentence.
  # Use "sequence_outputs" for token-level output.
  output_layer = bert_outputs["pooled_output"]

  hidden_size = output_layer.shape[-1].value #隐藏层

  # Create our own layer to tune for politeness data.
  output_weights = tf.get_variable(
      "output_weights", [num_labels, hidden_size],
      initializer=tf.truncated_normal_initializer(stddev=0.02)) #输出层网络权重

  output_bias = tf.get_variable(
      "output_bias", [num_labels], initializer=tf.zeros_initializer()) #输出层网络偏置

  with tf.variable_scope("loss"):

    # Dropout helps prevent overfitting
    output_layer = tf.nn.dropout(output_layer, keep_prob=0.9) #Dropout层

    logits = tf.matmul(output_layer, output_weights, transpose_b=True)
    logits = tf.nn.bias_add(logits, output_bias)
    log_probs = tf.nn.log_softmax(logits, axis=-1)

    # Convert labels into one-hot encoding
    one_hot_labels = tf.one_hot(labels, depth=num_labels, dtype=tf.float32) #label独热码

    predicted_labels = tf.squeeze(tf.argmax(log_probs, axis=-1, output_type=tf.int32))
    # If we're predicting, we want predicted labels and the probabiltiies.
    if is_predicting:
      return (predicted_labels, log_probs)

    # If we're train/eval, compute loss between predicted and actual label
    per_example_loss = -tf.reduce_sum(one_hot_labels * log_probs, axis=-1)
    loss = tf.reduce_mean(per_example_loss)
    return (loss, predicted_labels, log_probs)
```

Next we'll wrap our model function in a `model_fn_builder` function that adapts our model to work for training, evaluation, and prediction.

In [31]:

```
# model_fn_builder actually creates our model function
# using the passed parameters for num_labels, learning_rate, etc.
def model_fn_builder(num_labels, learning_rate, num_train_steps,
                     num_warmup_steps):
  """Returns `model_fn` closure for TPUEstimator."""
  def model_fn(features, labels, mode, params):  # pylint: disable=unused-argument
    """The `model_fn` for TPUEstimator."""

    input_ids = features["input_ids"]
    input_mask = features["input_mask"]
    segment_ids = features["segment_ids"]
    label_ids = features["label_ids"]

    is_predicting = (mode == tf.estimator.ModeKeys.PREDICT)

    # TRAIN and EVAL
    if not is_predicting: #非预测，训练阶段

      (loss, predicted_labels, log_probs) = create_model(
        is_predicting, input_ids, input_mask, segment_ids, label_ids, num_labels) #从预训练模型
中获取参数

      train_op = bert.optimization.create_optimizer(
          loss, learning_rate, num_train_steps, num_warmup_steps, use_tpu=False) #建优化器

      # Calculate evaluation metrics.
      def metric_fn(label_ids, predicted_labels): #评价指标
        accuracy = tf.metrics.accuracy(label_ids, predicted_labels)
        f1_score = tf.contrib.metrics.f1_score(
            label_ids,
            predicted_labels)
        auc = tf.metrics.auc(
            label_ids,
            predicted_labels)
        recall = tf.metrics.recall(
            label_ids,
            predicted_labels)
        precision = tf.metrics.precision(
            label_ids,
            predicted_labels)
        true_pos = tf.metrics.true_positives(
            label_ids,
            predicted_labels)
        true_neg = tf.metrics.true_negatives(
            label_ids,
            predicted_labels)
        false_pos = tf.metrics.false_positives(
            label_ids,
            predicted_labels)
        false_neg = tf.metrics.false_negatives(
            label_ids,
            predicted_labels)
        return {
            "eval_accuracy": accuracy,
            "f1_score": f1_score,
            "auc": auc,
            "precision": precision,
            "recall": recall,
            "true_positives": true_pos,
            "true_negatives": true_neg,
            "false_positives": false_pos,
            "false_negatives": false_neg
          }

      eval_metrics = metric_fn(label_ids, predicted_labels)

      if mode == tf.estimator.ModeKeys.TRAIN:
        return tf.estimator.EstimatorSpec(mode=mode,
          loss=loss,
          train_op=train_op)
      else:
          return tf.estimator.EstimatorSpec(mode=mode,
            loss=loss,
            eval_metric_ops=eval_metrics)
    else:
      (predicted_labels, log_probs) = create_model(
        is_predicting, input_ids, input_mask, segment_ids, label_ids, num_labels)

      predictions = {
          'probabilities': log_probs,
          'labels': predicted_labels
      }
      return tf.estimator.EstimatorSpec(mode, predictions=predictions)

  # Return the actual model function in the closure
  return model_fn
```

In [32]:

```
# Compute train and warmup steps from batch size
# These hyperparameters are copied from this colab notebook (https://colab.sandbox.google.com/gi
thub/tensorflow/tpu/blob/master/tools/colab/bert_finetuning_with_cloud_tpus.ipynb)
BATCH_SIZE = 32
LEARNING_RATE = 2e-5 #学习率非常小
NUM_TRAIN_EPOCHS = 3.0
# Warmup is a period of time where hte learning rate
# is small and gradually increases--usually helps training.
WARMUP_PROPORTION = 0.1
# Model configs
SAVE_CHECKPOINTS_STEPS = 500
SAVE_SUMMARY_STEPS = 100
```

In [33]:

```
# Compute # train and warmup steps from batch size
num_train_steps = int(len(train_features) / BATCH_SIZE * NUM_TRAIN_EPOCHS) #468 训练步数
num_warmup_steps = int(num_train_steps * WARMUP_PROPORTION) #46 预热步数
```

In [34]:

```
# Specify outpit directory and number of checkpoint steps to save
run_config = tf.estimator.RunConfig(
    model_dir=OUTPUT_DIR, #模型输出位置
    save_summary_steps=SAVE_SUMMARY_STEPS,
    save_checkpoints_steps=SAVE_CHECKPOINTS_STEPS)
```

In [36]:

```
model_fn = model_fn_builder(
  num_labels=len(label_list),
  learning_rate=LEARNING_RATE,
  num_train_steps=num_train_steps,
  num_warmup_steps=num_warmup_steps)

estimator = tf.estimator.Estimator(
  model_fn=model_fn,
  config=run_config,
  params={"batch_size": BATCH_SIZE})
```

```
INFO:tensorflow:Using config: {'_model_dir': 'OUTPUT_Bert', '_tf_random_seed': Non
e, '_save_summary_steps': 100, '_save_checkpoints_steps': 500, '_save_checkpoints_
secs': None, '_session_config': allow_soft_placement: true
graph_options {
  rewrite_options {
    meta_optimizer_iterations: ONE
  }
}
, '_keep_checkpoint_max': 5, '_keep_checkpoint_every_n_hours': 10000, '_log_step_c
ount_steps': 100, '_train_distribute': None, '_device_fn': None, '_protocol': Non
e, '_eval_distribute': None, '_experimental_distribute': None, '_experimental_max_
worker_delay_secs': None, '_session_creation_timeout_secs': 7200, '_service': Non
e, '_cluster_spec': <tensorflow.python.training.server_lib.ClusterSpec object at 0
x0000027D85682F60>, '_task_type': 'worker', '_task_id': 0, '_global_id_in_cluste
r': 0, '_master': '', '_evaluation_master': '', '_is_chief': True, '_num_ps_replic
as': 0, '_num_worker_replicas': 1}
```

```
INFO:tensorflow:Using config: {'_model_dir': 'OUTPUT_Bert', '_tf_random_seed': Non
e, '_save_summary_steps': 100, '_save_checkpoints_steps': 500, '_save_checkpoints_
secs': None, '_session_config': allow_soft_placement: true
graph_options {
  rewrite_options {
    meta_optimizer_iterations: ONE
  }
}
, '_keep_checkpoint_max': 5, '_keep_checkpoint_every_n_hours': 10000, '_log_step_c
ount_steps': 100, '_train_distribute': None, '_device_fn': None, '_protocol': Non
e, '_eval_distribute': None, '_experimental_distribute': None, '_experimental_max_
worker_delay_secs': None, '_session_creation_timeout_secs': 7200, '_service': Non
e, '_cluster_spec': <tensorflow.python.training.server_lib.ClusterSpec object at 0
x0000027D85682F60>, '_task_type': 'worker', '_task_id': 0, '_global_id_in_cluste
r': 0, '_master': '', '_evaluation_master': '', '_is_chief': True, '_num_ps_replic
as': 0, '_num_worker_replicas': 1}
```

Next we create an input builder function that takes our training feature set (`train_features`) and produces a generator. This is a pretty standard design pattern for working with Tensorflow Estimators (https://www.tensorflow.org/guide/estimators).

In [24]:

```
# Create an input function for training. drop_remainder = True for using TPUs.
train_input_fn = bert.run_classifier.input_fn_builder(
    features=train_features,
    seq_length=MAX_SEQ_LENGTH,
    is_training=True,
    drop_remainder=False)
```

Now we train our model! For me, using a Colab notebook running on Google's GPUs, my training time was about 14 minutes.

In [25]:

```
print('Beginning Training!')
current_time = datetime.now()
estimator.train(input_fn=train_input_fn, max_steps=num_train_steps)
print("Training took time ", datetime.now() - current_time)
```

```
Beginning Training!
WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\python\training\training_util.py:236: Variable.initialized_value (from
tensorflow.python.ops.variables) is deprecated and will be removed in a future ver
sion.
Instructions for updating:
Use Variable.read_value. Variables in 2.X are initialized automatically both in ea
ger and graph (inside tf.defun) contexts.

WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\python\training\training_util.py:236: Variable.initialized_value (from
tensorflow.python.ops.variables) is deprecated and will be removed in a future ver
sion.
Instructions for updating:
Use Variable.read_value. Variables in 2.X are initialized automatically both in ea
ger and graph (inside tf.defun) contexts.

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore

INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore

WARNING:tensorflow:From <ipython-input-18-ca03218f28a6>:34: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in
a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_p
rob`.

WARNING:tensorflow:From <ipython-input-18-ca03218f28a6>:34: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in
a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_p
rob`.

WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\bert
\optimization.py:27: The name tf.train.get_or_create_global_step is deprecated. Pl
ease use tf.compat.v1.train.get_or_create_global_step instead.


WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\bert
\optimization.py:27: The name tf.train.get_or_create_global_step is deprecated. Pl
ease use tf.compat.v1.train.get_or_create_global_step instead.


WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\bert
\optimization.py:32: The name tf.train.polynomial_decay is deprecated. Please use
tf.compat.v1.train.polynomial_decay instead.



WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\bert
\optimization.py:32: The name tf.train.polynomial_decay is deprecated. Please use
tf.compat.v1.train.polynomial_decay instead.
```

```
WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\bert
\optimization.py:70: The name tf.trainable_variables is deprecated. Please use tf.
compat.v1.trainable_variables instead.


WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\bert
\optimization.py:70: The name tf.trainable_variables is deprecated. Please use tf.
compat.v1.trainable_variables instead.


WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\python\ops\math_grad.py:1375: where (from tensorflow.python.ops.array_o
ps) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where

WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\python\ops\math_grad.py:1375: where (from tensorflow.python.ops.array_o
ps) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
f:\programming\anaconda3\envs\bert\lib\site-packages\tensorflow_core\python\framew
ork\indexed_slices.py:424: UserWarning: Converting sparse IndexedSlices to a dense
Tensor of unknown shape. This may consume a large amount of memory.
  "Converting sparse IndexedSlices to a dense Tensor of unknown shape. "

WARNING:tensorflow:
The TensorFlow contrib module will not be included in TensorFlow 2.0.
For more information, please see:
  * https://github.com/tensorflow/community/blob/master/rfcs/20180907-contrib-suns
et.md
  * https://github.com/tensorflow/addons
  * https://github.com/tensorflow/io (for I/O related ops)
If you depend on functionality not listed there, please file an issue.


WARNING:tensorflow:
The TensorFlow contrib module will not be included in TensorFlow 2.0.
For more information, please see:
  * https://github.com/tensorflow/community/blob/master/rfcs/20180907-contrib-suns
et.md
  * https://github.com/tensorflow/addons
  * https://github.com/tensorflow/io (for I/O related ops)
If you depend on functionality not listed there, please file an issue.


WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\contrib\metrics\python\metrics\classification.py:162: div (from tensorf
low.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Deprecated in favor of operator or tf.math.divide.

WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\contrib\metrics\python\metrics\classification.py:162: div (from tensorf
low.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Deprecated in favor of operator or tf.math.divide.

INFO:tensorflow:Done calling model_fn.

INFO:tensorflow:Done calling model_fn.

INFO:tensorflow:Create CheckpointSaverHook.
```

```
INFO:tensorflow:Create CheckpointSaverHook.

INFO:tensorflow:Graph was finalized.

INFO:tensorflow:Graph was finalized.

INFO:tensorflow:Restoring parameters from OUTPUT_Bert\model.ckpt-0

INFO:tensorflow:Restoring parameters from OUTPUT_Bert\model.ckpt-0

WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\python\training\saver.py:1069: get_checkpoint_mtimes (from tensorflow.p
ython.training.checkpoint_management) is deprecated and will be removed in a futur
e version.
Instructions for updating:
Use standard file utilities to get mtimes.

WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\python\training\saver.py:1069: get_checkpoint_mtimes (from tensorflow.p
ython.training.checkpoint_management) is deprecated and will be removed in a futur
e version.
Instructions for updating:
Use standard file utilities to get mtimes.

INFO:tensorflow:Running local_init_op.

INFO:tensorflow:Running local_init_op.

INFO:tensorflow:Done running local_init_op.

INFO:tensorflow:Done running local_init_op.

INFO:tensorflow:Saving checkpoints for 0 into OUTPUT_Bert\model.ckpt.

INFO:tensorflow:Saving checkpoints for 0 into OUTPUT_Bert\model.ckpt.

INFO:tensorflow:loss = 0.88821846, step = 1

INFO:tensorflow:loss = 0.88821846, step = 1

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 3 vs previous value: 3. You could i
ncrease the global step by passing tf.train.get_global_step() to Optimizer.apply_g
radients or Optimizer.minimize.

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 3 vs previous value: 3. You could i
ncrease the global step by passing tf.train.get_global_step() to Optimizer.apply_g
radients or Optimizer.minimize.

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 11 vs previous value: 11. You could
 increase the global step by passing tf.train.get_global_step() to Optimizer.apply_
gradients or Optimizer.minimize.

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 11 vs previous value: 11. You could
 increase the global step by passing tf.train.get_global_step() to Optimizer.apply_
gradients or Optimizer.minimize.

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 15 vs previous value: 15. You could
 increase the global step by passing tf.train.get_global_step() to Optimizer.apply_
gradients or Optimizer.minimize.
```

```
WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 15 vs previous value: 15. You could
increase the global step by passing tf.train.get_global_step() to Optimizer.apply_
gradients or Optimizer.minimize.

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 21 vs previous value: 21. You could
increase the global step by passing tf.train.get_global_step() to Optimizer.apply_
gradients or Optimizer.minimize.

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 21 vs previous value: 21. You could
increase the global step by passing tf.train.get_global_step() to Optimizer.apply_
gradients or Optimizer.minimize.

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 24 vs previous value: 24. You could
increase the global step by passing tf.train.get_global_step() to Optimizer.apply_
gradients or Optimizer.minimize.

WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not be
en increased. Current value (could be stable): 24 vs previous value: 24. You could
increase the global step by passing tf.train.get_global_step() to Optimizer.apply_
gradients or Optimizer.minimize.

INFO:tensorflow:global_step/sec: 0.0809963

INFO:tensorflow:global_step/sec: 0.0809963

INFO:tensorflow:loss = 0.34584528, step = 101 (1234.233 sec)

INFO:tensorflow:loss = 0.34584528, step = 101 (1234.233 sec)

INFO:tensorflow:global_step/sec: 0.0883234

INFO:tensorflow:global_step/sec: 0.0883234

INFO:tensorflow:loss = 0.1824758, step = 201 (1132.074 sec)

INFO:tensorflow:loss = 0.1824758, step = 201 (1132.074 sec)

INFO:tensorflow:global_step/sec: 0.0891647

INFO:tensorflow:global_step/sec: 0.0891647

INFO:tensorflow:loss = 0.08375876, step = 301 (1121.463 sec)

INFO:tensorflow:loss = 0.08375876, step = 301 (1121.463 sec)

INFO:tensorflow:global_step/sec: 0.0866038

INFO:tensorflow:global_step/sec: 0.0866038

INFO:tensorflow:loss = 0.00814178, step = 401 (1154.731 sec)

INFO:tensorflow:loss = 0.00814178, step = 401 (1154.731 sec)

INFO:tensorflow:Saving checkpoints for 468 into OUTPUT_Bert\model.ckpt.

INFO:tensorflow:Saving checkpoints for 468 into OUTPUT_Bert\model.ckpt.

INFO:tensorflow:Loss for final step: 0.004126965.

INFO:tensorflow:Loss for final step: 0.004126965.

Training took time  1:35:10.964661
```

Now let's use our test data to see how well our model did:

In [26]:

```
test_input_fn = run_classifier.input_fn_builder(
    features=test_features,
    seq_length=MAX_SEQ_LENGTH,
    is_training=False,
    drop_remainder=False)
```

In [27]:

```
estimator.evaluate(input_fn=test_input_fn, steps=None)
```

```
INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore

INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore
f:\programming\anaconda3\envs\bert\lib\site-packages\tensorflow_core\python\framew
ork\indexed_slices.py:424: UserWarning: Converting sparse IndexedSlices to a dense
Tensor of unknown shape. This may consume a large amount of memory.
  "Converting sparse IndexedSlices to a dense Tensor of unknown shape. "

INFO:tensorflow:Done calling model_fn.

INFO:tensorflow:Done calling model_fn.

INFO:tensorflow:Starting evaluation at 2019-11-17T17:01:17Z

INFO:tensorflow:Starting evaluation at 2019-11-17T17:01:17Z

INFO:tensorflow:Graph was finalized.

INFO:tensorflow:Graph was finalized.

INFO:tensorflow:Restoring parameters from OUTPUT_Bert\model.ckpt-468

INFO:tensorflow:Restoring parameters from OUTPUT_Bert\model.ckpt-468

INFO:tensorflow:Running local_init_op.

INFO:tensorflow:Running local_init_op.

INFO:tensorflow:Done running local_init_op.

INFO:tensorflow:Done running local_init_op.

INFO:tensorflow:Finished evaluation at 2019-11-17-17:12:18

INFO:tensorflow:Finished evaluation at 2019-11-17-17:12:18

INFO:tensorflow:Saving dict for global step 468: auc = 0.8657599, eval_accuracy =
0.8658, f1_score = 0.86736506, false_negatives = 324.0, false_positives = 347.0, g
lobal_step = 468, loss = 0.5052588, precision = 0.8634396, recall = 0.87132645, tr
ue_negatives = 2135.0, true_positives = 2194.0

INFO:tensorflow:Saving dict for global step 468: auc = 0.8657599, eval_accuracy =
0.8658, f1_score = 0.86736506, false_negatives = 324.0, false_positives = 347.0, g
lobal_step = 468, loss = 0.5052588, precision = 0.8634396, recall = 0.87132645, tr
ue_negatives = 2135.0, true_positives = 2194.0

INFO:tensorflow:Saving 'checkpoint_path' summary for global step 468: OUTPUT_Bert
\model.ckpt-468

INFO:tensorflow:Saving 'checkpoint_path' summary for global step 468: OUTPUT_Bert
\model.ckpt-468
```

Out[27]:

```
{'auc': 0.8657599,
 'eval_accuracy': 0.8658,
 'f1_score': 0.86736506,
 'false_negatives': 324.0,
 'false_positives': 347.0,
 'loss': 0.5052588,
 'precision': 0.8634396,
 'recall': 0.87132645,
 'true_negatives': 2135.0,
 'true_positives': 2194.0,
 'global_step': 468}
```

Now let's write code to make predictions on new sentences:

In [23]:

```
def getPrediction(in_sentences):
  labels = ["Negative", "Positive"]
  input_examples = [run_classifier.InputExample(guid="", text_a = x, text_b = None, label = 0) for x in in_sentences] # here, "" is just a dummy label
  input_features = run_classifier.convert_examples_to_features(input_examples, label_list, MAX_SEQ_LENGTH, tokenizer)
  predict_input_fn = run_classifier.input_fn_builder(features=input_features, seq_length=MAX_SEQ_LENGTH, is_training=False, drop_remainder=False)
  predictions = estimator.predict(predict_input_fn)
  return [(sentence, prediction['probabilities'], labels[prediction['labels']]) for sentence, prediction in zip(in_sentences, predictions)]
```

In [24]:

```
pred_sentences = [
  "That movie was absolutely awful",
  "The acting was a bit lacking",
  "The film was creative and surprising",
  "Absolutely fantastic!"
]
```

In [30]:

```
predictions = getPrediction(pred_sentences)
```

```
INFO:tensorflow:Writing example 0 of 4

INFO:tensorflow:Writing example 0 of 4

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid:

INFO:tensorflow:guid:

INFO:tensorflow:tokens: [CLS] that movie was absolutely awful [SEP]

INFO:tensorflow:tokens: [CLS] that movie was absolutely awful [SEP]

INFO:tensorflow:input_ids: 101 2008 3185 2001 7078 9643 102 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_ids: 101 2008 3185 2001 7078 9643 102 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid:

INFO:tensorflow:guid:

INFO:tensorflow:tokens: [CLS] the acting was a bit lacking [SEP]

INFO:tensorflow:tokens: [CLS] the acting was a bit lacking [SEP]

INFO:tensorflow:input_ids: 101 1996 3772 2001 1037 2978 11158 102 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:input_ids: 101 1996 3772 2001 1037 2978 11158 102 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid:

INFO:tensorflow:guid:

INFO:tensorflow:tokens: [CLS] the film was creative and surprising [SEP]

INFO:tensorflow:tokens: [CLS] the film was creative and surprising [SEP]

INFO:tensorflow:input_ids: 101 1996 2143 2001 5541 1998 11341 102 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_ids: 101 1996 2143 2001 5541 1998 11341 102 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid:

INFO:tensorflow:guid:

INFO:tensorflow:tokens: [CLS] absolutely fantastic ! [SEP]

INFO:tensorflow:tokens: [CLS] absolutely fantastic ! [SEP]

INFO:tensorflow:input_ids: 101 7078 10392 999 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_ids: 101 7078 10392 999 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore

INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore

INFO:tensorflow:Done calling model_fn.

INFO:tensorflow:Done calling model_fn.
```

```
INFO:tensorflow:Graph was finalized.

INFO:tensorflow:Graph was finalized.

INFO:tensorflow:Restoring parameters from OUTPUT_Bert\model.ckpt-468

INFO:tensorflow:Restoring parameters from OUTPUT_Bert\model.ckpt-468

INFO:tensorflow:Running local_init_op.

INFO:tensorflow:Running local_init_op.

INFO:tensorflow:Done running local_init_op.

INFO:tensorflow:Done running local_init_op.
```

Voila! We have a sentiment classifier!

In [31]:

```
predictions
```

Out[31]:

```
[('That movie was absolutely awful',
  array([-1.7498910e-03, -6.3490496e+00], dtype=float32),
  'Negative'),
 ('The acting was a bit lacking',
  array([-0.00662121, -5.0207834 ], dtype=float32),
  'Negative'),
 ('The film was creative and surprising',
  array([-5.9929047e+00, -2.4995534e-03], dtype=float32),
  'Positive'),
 ('Absolutely fantastic!',
  array([-5.4728165e+00, -4.2082807e-03], dtype=float32),
  'Positive')]
```

使用上次训练好的模型进行预测 （训练好模型后，给其他同事直接使用）

In [1]:

```
#加载训练好的模型
import tensorflow as tf
with tf.Session() as sess:
  new_saver = tf.train.import_meta_graph('E:\\GoogleDrive\\XJTLU-PhD\\bert-master\\OUTPUT_Bert
\\model.ckpt-468.meta')
  new_saver.restore(sess, tf.train.latest_checkpoint('E:\\GoogleDrive\\XJTLU-PhD\\bert-master\\
OUTPUT_Bert\\'))
```

```
INFO:tensorflow:Restoring parameters from E:\GoogleDrive\XJTLU-PhD\bert-master\OUT
PUT_Bert\model.ckpt-468
```

In [37]:

```
#直接使用
predictions = getPrediction(pred_sentences)
```

```
INFO:tensorflow:Writing example 0 of 4

INFO:tensorflow:Writing example 0 of 4

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***


INFO:tensorflow:guid:

INFO:tensorflow:guid:

INFO:tensorflow:tokens: [CLS] that movie was absolutely awful [SEP]

INFO:tensorflow:tokens: [CLS] that movie was absolutely awful [SEP]

INFO:tensorflow:input_ids: 101 2008 3185 2001 7078 9643 102 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_ids: 101 2008 3185 2001 7078 9643 102 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid:

INFO:tensorflow:guid:

INFO:tensorflow:tokens: [CLS] the acting was a bit lacking [SEP]

INFO:tensorflow:tokens: [CLS] the acting was a bit lacking [SEP]

INFO:tensorflow:input_ids: 101 1996 3772 2001 1037 2978 11158 102 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:input_ids: 101 1996 3772 2001 1037 2978 11158 102 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid:

INFO:tensorflow:guid:

INFO:tensorflow:tokens: [CLS] the film was creative and surprising [SEP]

INFO:tensorflow:tokens: [CLS] the film was creative and surprising [SEP]

INFO:tensorflow:input_ids: 101 1996 2143 2001 5541 1998 11341 102 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_ids: 101 1996 2143 2001 5541 1998 11341 102 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)


INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid:

INFO:tensorflow:guid:

INFO:tensorflow:tokens: [CLS] absolutely fantastic ! [SEP]

INFO:tensorflow:tokens: [CLS] absolutely fantastic ! [SEP]

INFO:tensorflow:input_ids: 101 7078 10392 999 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_ids: 101 7078 10392 999 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:label: 0 (id = 0)

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore

INFO:tensorflow:Saver not created because there are no variables in the graph to r
estore
```

```
WARNING:tensorflow:From <ipython-input-30-ca03218f28a6>:34: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in
a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_p
rob`.

WARNING:tensorflow:From <ipython-input-30-ca03218f28a6>:34: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in
a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_p
rob`.

INFO:tensorflow:Done calling model_fn.

INFO:tensorflow:Done calling model_fn.

WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\python\ops\array_ops.py:1475: where (from tensorflow.python.ops.array_o
ps) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where

WARNING:tensorflow:From f:\programming\anaconda3\envs\bert\lib\site-packages\tenso
rflow_core\python\ops\array_ops.py:1475: where (from tensorflow.python.ops.array_o
ps) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where

INFO:tensorflow:Graph was finalized.

INFO:tensorflow:Graph was finalized.

INFO:tensorflow:Restoring parameters from OUTPUT_Bert\model.ckpt-468

INFO:tensorflow:Restoring parameters from OUTPUT_Bert\model.ckpt-468

INFO:tensorflow:Running local_init_op.

INFO:tensorflow:Running local_init_op.

INFO:tensorflow:Done running local_init_op.

INFO:tensorflow:Done running local_init_op.
```

In [38]:

```
predictions
```

Out[38]:

```
[('That movie was absolutely awful',
  array([-1.9136227e-03, -6.2597380e+00], dtype=float32),
  'Negative'),
 ('The acting was a bit lacking',
  array([-0.00731551, -4.9214096 ], dtype=float32),
  'Negative'),
 ('The film was creative and surprising',
  array([-6.1452498e+00, -2.1459663e-03], dtype=float32),
  'Positive'),
 ('Absolutely fantastic!',
  array([-5.5189867e+00, -4.0179724e-03], dtype=float32),
  'Positive')]
```

Reference

NLP从词袋到Word2Vec的文本表示 (https://juejin.im/post/5cd41afa6fb9a032332b47a4)
https://www.cnblogs.com/rucwxb/p/10367609.html (https://www.cnblogs.com/rucwxb/p/10367609.html)
https://www.cnblogs.com/rucwxb/p/10277217.html (https://www.cnblogs.com/rucwxb/p/10277217.html)
从Word Embedding到Bert模型—自然语言处理中的预训练技术发展史 张俊林
(https://zhuanlan.zhihu.com/p/49271699)
【NLP】Attention Model（注意力模型）学习总结 (https://www.cnblogs.com/guoyaohua/p/9429924.html)

拓展阅读

分布式表示

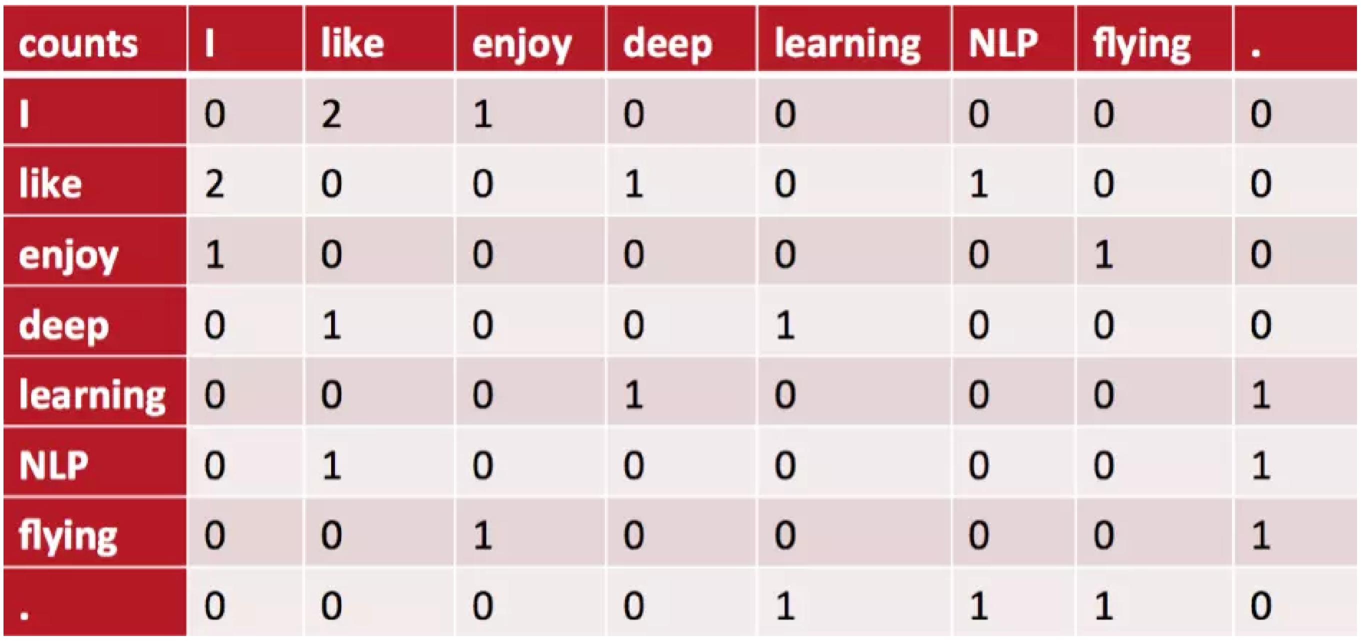
用一个词附近的其它词来表示该词，这是现代统计自然语言处理中最有创见的想法之一。 当初科学家发明这种方法是基于人的语言表达，认为一个词是由这个词的周边词汇一起来构成精确的语义信息。就好比，物以类聚人以群分，如果你想了解一个人，可以通过他周围的人进行了解，因为周围人都有一些共同点才能聚集起来。

- 共现矩阵 共现矩阵顾名思义就是共同出现的意思，词文档的共现矩阵主要用于发现主题(topic)，用于主题模型，如LSA。

局域窗中的word-word共现矩阵可以挖掘语法和语义信息，例如：

I like deep learning.
I like NLP.
I enjoy flying

有以上三句话，设置滑窗为2，可以得到一个词典：{"I like","like deep","deep learning","like NLP","I enjoy","enjoy flying","I like"}。 我们可以得到一个共现矩阵(对称矩阵)：



counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

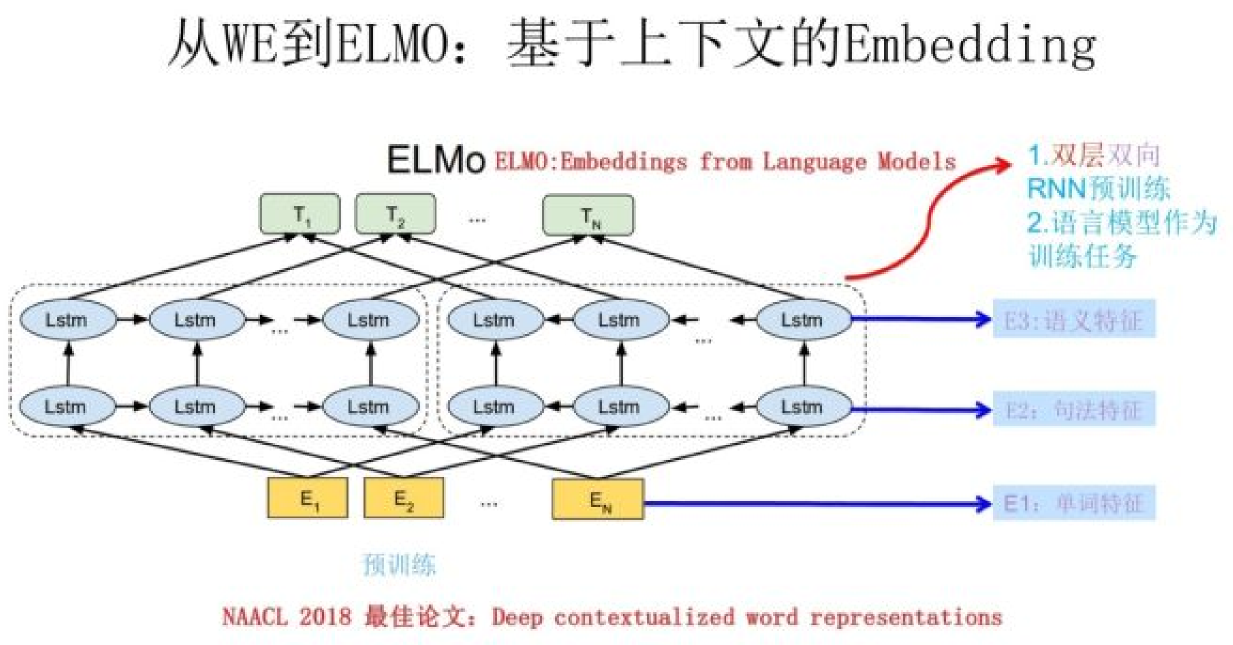
中间的每个格子表示的是行和列组成的词组在词典中共同出现的次数，也就体现了共现的特性。

存在的问题：

- 向量维数随着词典大小线性增长。
- 存储整个词典的空间消耗非常大。
- 一些模型如文本分类模型会面临稀疏性问题。
- 模型会欠稳定，每新增一份语料进来，稳定性就会变化。

从Word Embedding到ELMO

ELMO是“Embedding from Language Models”的简称，其实这个名字并没有反应它的本质思想，提出ELMO的论文题目：“Deep contextualized word representation”更能体现其精髓，而精髓在哪里？在deep contextualized这个短语，一个是deep，一个是context，其中context更关键。在此之前的Word Embedding本质上是个静态的方式，所谓静态指的是训练好之后每个单词的表达就固定住了，以后使用的时候，不论新句子上下文单词是什么，这个单词的Word Embedding不会跟着上下文场景的变化而改变，所以对于比如Bank这个词，它事先学好的Word Embedding中混合了几种语义，在应用中来了个新句子，即使从上下文中（比如句子包含money等词）明显可以看出它代表的是“银行”的含义，但是对应的Word Embedding内容也不会变，它还是混合了多种语义。这是为何说它是静态的，这也是问题所在。ELMO的本质思想是：我事先用语言模型学好一个单词的Word Embedding，此时多义词无法区分，不过这没关系。在我实际使用Word Embedding的时候，单词已经具备了特定的上下文了，这个时候我可以根据上下文单词的语义去调整单词的Word Embedding表示，这样经过调整后的Word Embedding更能表达在这个上下文中的具体含义，自然也就解决了多义词的问题了。所以ELMO本身是个根据当前上下文对Word Embedding动态调整的思路。


从WE到ELMO：基于上下文的Embedding
ELMo ELMO:Embeddings from Language Models
1.双层双向
RNN预训练
2.语言模型作为
训练任务
T1
T2
TN
Lstm
E3:语义特征
E2：句法特征
E1：单词特征
E1
E2
EN
预训练
NAACL 2018 最佳论文：Deep contextualized word representations