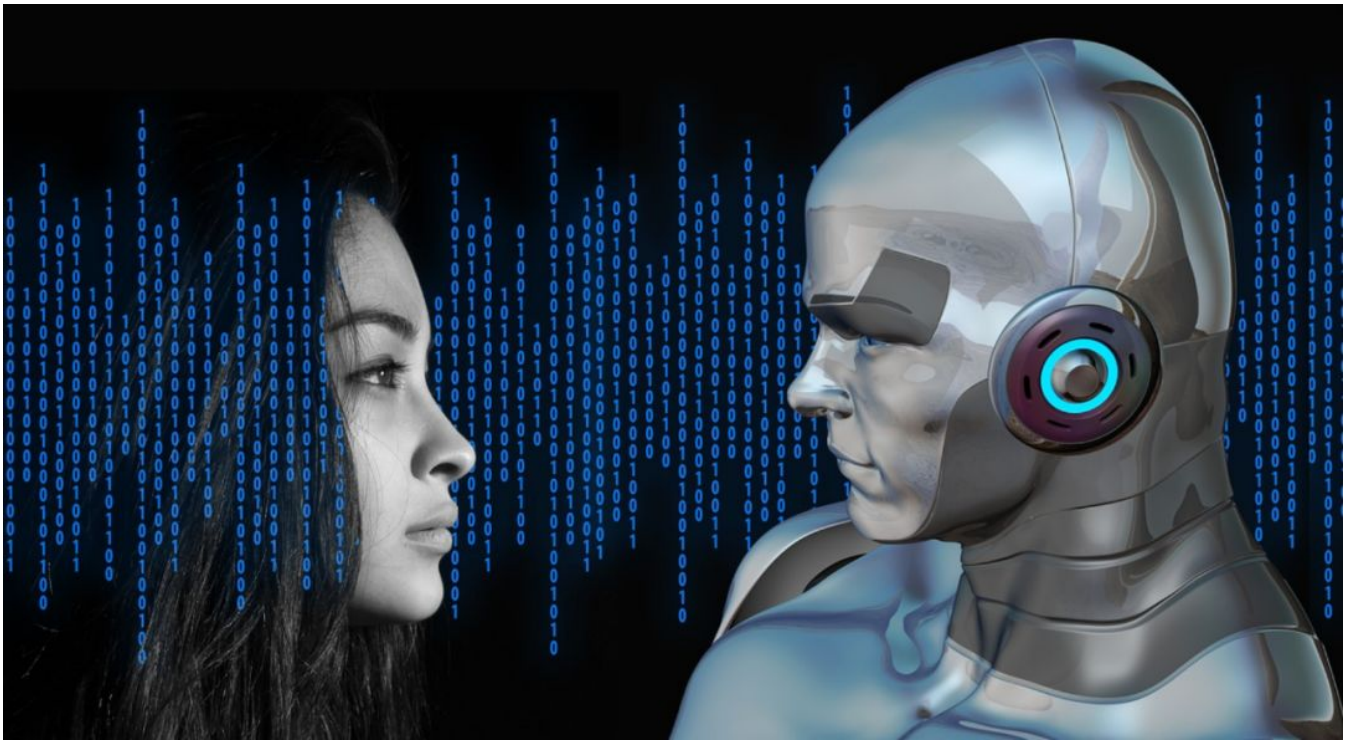
女人的嘴，骗人的鬼？ELMo教你用算法分辨女人心

原创 读芯术 读芯术 2019-04-09

全文共**8884**字，预计学习时长**23**分钟



“语境”有多重要呢？

比如：你写了一篇论文放在桌上，一般人问你：“这是什么东西？”那可能只是纯粹地问这是什么东东。但如果是你的导师问你，那问题就严重了，潜台词是：这TM写的是什么玩意儿？

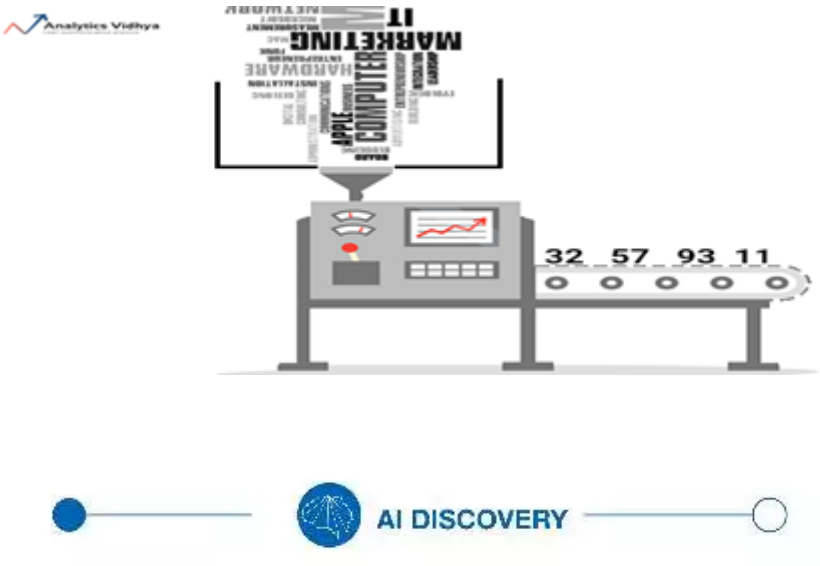
语言有着人类最复杂的游戏法则。尤其是在恋爱关系中，女孩子往往游刃有余，她们一时兴起笑里藏刀抛出的问题，往往让男生分分钟“送命”，比如这样的：

没有男友的小编只能调戏Siri，它的回答是这样的：

好吧，“直男”中的战斗机，不配在爱情界拥有姓名！

机器无法理解句子的真正含义，一直是NLP从业者心中的一根刺。传统的NLP技术和框架只能满足基本的语言任务。但当试图增加语境时，形势便急转而下。

不过，近18个月以来，NLP的形势发生了显著变化。像 Google的BERT和Zalando的 Flair这样的NLP框架能够通过句子进行解析，并掌握它们的语境。



语言模型嵌入（ELMo）

在这方面最大的突破之一来自于Elmo，一个由AllenNLP开发的最先进的NLP框架。本文，我们将探讨ELMo（语言模型嵌入），并使用它在真实数据集上用Python构建一个令人兴奋的NLP模型。

目录

1.什么是ELMo?
2.了解ELMo的工作原理
3.ELMo与其他单词嵌入有什么不同?
4.实现：ELMO用于Python中的文本分类
    4.1 理解问题陈述
    4.2 关于数据集
    4.3 导入库
    4.4 读取和检查数据
    4.5 文本清理及预处理
    4.6 TensorFlow Hub简介
    4.7 准备ELMO向量
    4.8 建模与评估
5.ELMo的其他用途是什么?

ELMo是什么?

不，这里的ELMo不是SesameStreet中的人物！而是一种在向量或嵌入中表示单词的新方法。这些单词嵌入有助于在一些NLP任务中实现最先进的（SOTA）结果：

Task	Previous SOTA		ELMo + Baseline
SQuAD	SAN	84.4	85.8
SNLI	Chen et al (2017)	88.6	88.7 +/- 0.17
SRL	He et al (2017)	81.7	84.6
Coref	Lee et al (2017)	67.2	70.4
NER	Peters et al (2017)	91.93 +/- 0.19	92.22 +/- 0.10
Sentiment (5-class)	McCann et al (2017)	53.7	54.7 +/- 0.5

全球的NLP科学家已经开始将ELMo用于各种NLP任务，包括研究和工业。

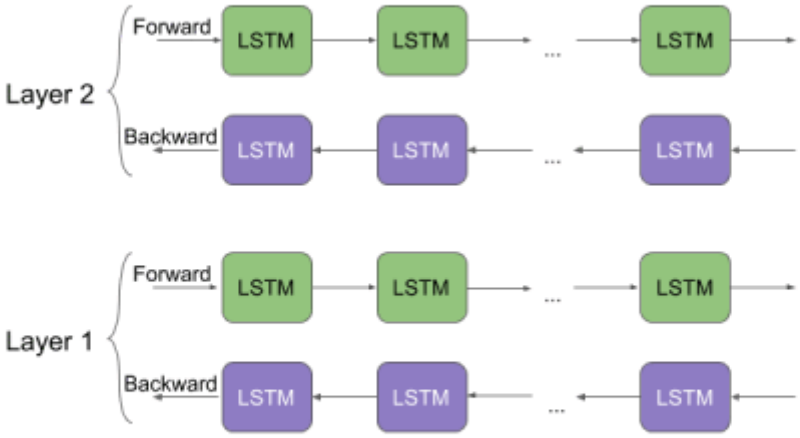
了解ELMo的工作原理

在用Python实现ELMo之前，让我们先直观了解一下它是如何工作的。

想象一下：你已经成功地将ELMo代码从GitHub复制到了Python中，并在自定义文本数据上构建了一个模型。你得到了平均的结果，所以现在需要改进模型。如果不了解ELMo的架构，你会怎么做？如果没有研究过，你会调整哪些参数？

这一思路适用于所有机器学习算法。你不需要了解其他分支，但是你应该具备足够的知识来将其用于模型改进中。现在，让我们回到ELMo的工作方式。

ELMo字向量是在两层双向语言模型（biLM）之上计算的。这个biLM模型有两层堆叠在一起。每层有2个通道-前向通道和后向通道：



- 上述架构使用字符级卷积神经网络（CNN）将文本字符串中的单词表示为原始单词向量。
- 这些原始单词向量作为biLM第一层的输入。
- 前向通道包含关于某个单词及该单词之前语境（其他词）的信息
- 后向通道包含有关单词及该单词之后语境的信息
- 来自前向通道和后向通道的这对信息，形成中间词向量。
- 这些中间字向量被送入biLM的下一层
- （ELMO的）最终表示是原始单词向量和两个中间字向量的加权和。

由于biLM的输入是根据字符而不是单词来计算的，因此它获得了单词的内部结构。例如，biLM可以理解类似beauty和beautiful这样的术语某种程度上是相关的，甚至不需要考虑它们经常出现的语境。听起来太不可思议了！

ELMo和其他单词嵌入有什么不同?

与传统的单词嵌入（如word2vec和GLoVe）不同，分配给标记或单词的ELMo向量实际上是包含该单词的整个句子的函数。因此，在不同的语境下，同一个单词可以有不同的词向量。

也许你们会问：知道这些如何帮助处理NLP问题？让我用一个例子来解释这一点。

假设有以下几个句子：

1. *Iread the book yesterday.*

2. Canyou read the letter now?

花点时间思考一下这两者之间的区别。第一句中的动词“read”是过去式。同一个动词在第二句中转换成现在时态。这是一种一词多义现象，一个词可以有多种含义或意义。

语言是如此的复杂。

传统的单词嵌入为两个句子中的单词“read”提供了相同的向量。因此，该系统无法区分多义词。这些单词嵌入无法掌握单词使用的语境。

ELMo 单词向量成功解决了这个问题。ELMo单词表示法将整个输入语句转化为公式，用于计算单词嵌入。因此，“read”一词在不同的语境中具有不同的ELMo向量。

实现：用于Python中文本分类的ELMo

现在是你一直在等待的时刻——在Python中实现ELMo！让我们一步一步来。

1.理解问题陈述

处理任何数据科学挑战的第一步是定义问题陈述。这是我们未来行动的基础。

对于本文，我们已经准备好了问题陈述：

情感分析仍然是自然语言处理（NLP）广泛应用的关键问题之一。这一次，考虑到客户关于制造和销售手机、电脑、笔记本电脑等的各种科技公司的tweet，任务是确定tweet是否对这些公司或产品有负面情绪。

这显然是一个二元文本分类任务，其中我们必须从摘取的推特中预测情感。

2.关于数据集

以下是我们所拥有的数据集的分类：

•训练集包含7920条推特
•测试集包含1953条推特

你可以从此页下载数据集。请注意，必须注册或登录才能下载。

警告：推特中的大多数亵渎和粗俗词汇已被替换为“$&@*#”。但是，请注意，数据集可能仍然包含可能被认为是亵渎、粗俗或冒犯的文本。

好吧，让我们启动最喜欢的Python IDE并进行编码！

3.导入库

导入将在notebook中使用的库：

```
importpandas as pd

        importnumpy as np

        importspacy

        fromtqdm import tqdm

        importre

        importtime

        importpickle

        pd.set_option('display.max_colwidth',200)

```

4.读取并检查数据

```
# read data
train =pd.read_csv("train_2kmZucJ.csv")
test =pd.read_csv("test_oJQbWVk.csv")
train.shape, test.shape
```

输出: ((7920, 3), (1953, 2))

训练集有7920条推特，而测试组只有1953条。现在检查一下训练集中的类别分布：

```
train['label'].value_counts(normalize =True)
```

输出:
0 0.744192
1 0.255808
Name: label, dtype: float64

在这里，“1”表示否定的tweet，而“0”表示非否定的tweet。
快速浏览一下训练集的前5行：

```
train.head()
```

	id	label	tweet
0	1	0	#fingerprint #Pregnancy Test https://goo.gl/h1MfQV #android #apps #beautiful #cute #health #igers #iphoneonly #iphonesia #iphone
1	2	0	Finally a transparant silicon case ^^ Thanks to my uncle :) #yay #Sony #Xperia #S #sonyexperias... http://instagram.com/p/YGEt5JC6JM/
2	3	0	We love this! Would you go? #talk #makememories #unplug #relax #iphone #smartphone #wifi #connect... http://fb.me/6N3LsUpCu
3	4	0	I'm wired I know I'm George I was made that way ;) #iphone #cute #daventry #home http://instagr.am/p/Li_5_ujS4k/
4	5	1	What amazing service! Apple won't even talk to me about a question I have unless I pay them $19.95 for their stupid support!

我们有三列要处理。“tweet”列是独立变量，而“label”列是目标变量。

5．文本清洗和预处理

我们将拥有一个干净、结构化的数据集，以便在理想情况下使用。但是在NLP中事情还没有那么简单。

我们需要花费大量的时间清理数据，以便为模型构建阶段做好准备。从文本中提取特征较为容易，甚至特征中包含更多信息。数据质量变得越高，模型的性能的改善越有意义。

所以，让我们清理一下收到的文本，并进行探索。

在推特中似乎有相当多的URL链接。他们没有告诉我们太多（如果有的话）关于推特的情感，所以将其直接删除。

```
        #remove URL's from train and test

        train['clean_tweet']= train['tweet'].apply(lambda x: re.sub(r'http\S+',

        test['clean_tweet']= test['tweet'].apply(lambda x: re.sub(r'http\S+', ''
```

我们使用正则表达式（或RegEx）来删除URL。

注意：你可以在这篇文章中了解更多关于Regex的信息。

现在让我们来做一些常规的文字清理。

```
#remove punctuation marks

        punctuation= '!"#$%&()*+-/:;<=>?@[\\]^_`{|}~'



        train['clean_tweet']= train['clean_tweet'].apply(lambda x: ''.join(ch f

        test['clean_tweet']= test['clean_tweet'].apply(lambda x: ''.join(ch for



        #convert text to lowercase

        train['clean_tweet']= train['clean_tweet'].str.lower()

```

```
        test['clean_tweet']= test['clean_tweet'].str.lower()



        #remove numbers

        train['clean_tweet']= train['clean_tweet'].str.replace("[0-9]", " ")

        test['clean_tweet']= test['clean_tweet'].str.replace("[0-9]", " ")



        #remove whitespaces

        train['clean_tweet']= train['clean_tweet'].apply(lambda x:' '.join(x.sp

        test['clean_tweet']= test['clean_tweet'].apply(lambda x: ' '.join(x.spl
```

我还想对文本进行规范化，即执行文本规范化。这有助于将单词缩减为其基本形式。例如，单词'produces'、'production'和'producing'的基本形式是'product'。通常情况下，同一单词的多种形式并不那么重要，我们只需要知道该单词的基本形式。

我们将利用流行的spacy库将文本按屈折变化形式进行归类（标准化）。

```
#import spaCy's language model

        nlp= spacy.load('en', disable=['parser', 'ner'])



        #function to lemmatize text

        deflemmatization(texts):

            output = []

```

```
        for i in texts:

                s = [token.lemma_ for token innlp(i)]

            output.append(' '.join(s))

        return output
```

标准化训练集和测试集中的tweet：

```
train['clean_tweet'] =lemmatization(train['clean_tweet'])

test['clean_tweet'] =lemmatization(test['clean_tweet'])
```

让我们快速浏览一下原始推特和清理过的推特：

```
train.sample(10)
```

	id	label	tweet	clean_tweet
3802	3803	0	My father is taking me out to buy me my first gaming console ever - #PS4. #speechless #firsttimeever #sony #playstation	-PRON- father be take -PRON- out to buy -PRON- -PRON- first gaming console ever ps . speechless firsttimeever sony playstation
1314	1315	0	My morning #Toronto #coffee #cup #cheesecake #yum #breakfest #Apple #MacBook #girl #morning … http://instagram.com/p/sVGp2rP51_/	-PRON- morning toronto coffee cup cheesecake yum breakf apple macbook girl morning …
1289	1290	1	Why is there always a new version of iTunes to download?!	why be there always a new version of itune to download
7787	7788	0	So true. Plus when I changed from Macbook Pro to Imac I find that Iphoto and Aperture are no longer available and a kids programme called Photos designed for I know not whom is substituted with no...	so true . plus when i change from macbook pro to imac i find that iphoto and aperture be no longer available and a kid programme call photo design for i know not whom be substitute with no alterna...
6579	6580	0	Sooo loving my #samsung galaxy s3....cant seem to put it down.	sooo love -PRON- samsung galaxy s .... can not seem to put -PRON- down .
2646	2647	0	Gain Followers RT This MUST FOLLOW ME I FOLLOW BACK Follow everyone who rts Gain #iphone #sougofollow +rz6	gain follower rt this must follow -PRON- i follow back follow everyone who rt gain iphone sougofollow rz
3534	3535	0	1 week with the #samsung #note3 ... loving it so far! Not as big as I thought it would be... #androidpic.twitter.com/OPWk5Jgtec	week with the samsung note ... love -PRON- so far not as big as i think -PRON- would be ... androidpic.twitter.comopwk jgtec
3027	3028	0	If Microsoft had anything even close to the IPod Id be the first one to switch over #worstcustomerservice	if microsoft have anything even close to the ipod -PRON- would be the first one to switch over worstcustomerservice
4852	4853	0	Yes! Finally got the latest snapchat update! Now I get the dancing ghost just like everyone else! #samsung #galaxy @SnapchatProbbz	yes finally get the late snapchat update now i get the dancing ghost just like everyone else samsung galaxy snapchatprobbz
7474	7475	1	@gdavidson88 fannies. Don't they know #apple #products and that #Steve #Jobs was #KKK ??? Sent from my iPhone	gdavidson fanny . do not -PRON- know apple product and that steve job be kkk send from -PRON- iphone

仔细查看以上列。“clean_tweet”栏中的tweet看起来比原来的tweet更易读。

但是，我觉得仍然有足够的空间来清理文本。我鼓励您尽可能多地探索数据，并在文本中找到更多的见解或不规则之处。

6、TensorFlow Hub简介

等等，TensorFlow和我们的教程有什么关系？

TensorFlow Hub是一个库，通过允许为不同的任务使用许多机器学习模型来实现转移学习。ELMo就是这样一个例子。这就是为什么我们将在实现中通过TensorFlow Hub访问ELMO。

在进行其他步骤之前，需要安装TensorFlow Hub。要使用TensorFlow Hub，必须安装TensorFlow软件包或将其升级到至少1.7版本：

```
$ pip install"tensorflow>=1.7.0"

$ pip install tensorflow-hub
```

7．准备ELMo向量

现在将导入经过预训练的ELMo模型。请注意：该模型的大小超过350MB，因此你可能需要一段时间来下载。

```
import tensorflow_hub as hub

import tensorflow as tf

elmo =hub.Module("https://tfhub.dev/google/elmo/2", trainable=True)
```

我将首先展示如何获得一个句子的ELMo向量。你所要做的就是在对象elmo中传递一个字符串列表。

```
#just a random sentence

        x= ["Roasted ants are a popular snack in Columbia"]



        #Extract ELMo features

        embeddings= elmo(x, signature="default", as_dict=True)["elmo"]



        embeddings.shape

```

输出: TensorShape([Dimension(1),Dimension(8), Dimension(1024)])

输出是一个三维形状的张量（1，8，1024）：

•该张量的第一维表示训练样本的数量。在我们的示例中是1
•第二个维度表示字符串输入列表中最长字符串的最大长度。因为输入列表中只有一个字符串，所以第二个维度的大小等于字符串的长度–8
•第三个维度等于ELMo向量的长度

因此，输入语句中的每个单词都有一个大小为1024的ELMo向量。

让我们继续为训练和测试数据集中清理过的推特提取ELMo向量。然而，为了得到整个推特的向量表示，我们将取推特的组成项或标记的ELMo向量的平均值。

让我们定义一个函数来执行此操作：

```
defelmo_vectors(x):

        embeddings = elmo(x.tolist(),signature="default", as_dict=True)["elmo

        with tf.Session() as sess:

          sess.run(tf.global_variables_initializer())

          sess.run(tf.tables_initializer())

          # return average of ELMo features

          returnsess.run(tf.reduce_mean(embeddings,1))
```

如果使用上面的函数一次性提取推特的嵌入，可能会耗尽计算资源（内存）。一种解决方法是：将训练和测试集分成每批100个样本，然后，将这些批次依次传递给elmo_vectors( )函数

我将把这些批次保存在一个列表中：

```
list_train = [train[i:i+100] for i inrange(0,train.shape[0],100)]

list_test = [test[i:i+100] for i inrange(0,test.shape[0],100)]
```

现在，我们将遍历这些批次并提取ELMO向量。友情提示，这要花很长时间。

```
# Extract ELMo embeddings

elmo_train = [elmo_vectors(x['clean_tweet'])for x in list_train]

elmo_test = [elmo_vectors(x['clean_tweet'])for x in list_test]
```

一旦拥有了所有的向量，就可以将它们连接成一个数组：

```
elmo_train_new = np.concatenate(elmo_train,axis = 0)

elmo_test_new = np.concatenate(elmo_test,axis = 0)
```

我建议对这些数组进行保存，因为我们花了很长时间为其获取ELMo向量。我们将其保存为pickle文件：

```
 #save elmo_train_new

       pickle_out= open("elmo_train_03032019.pickle","wb")

       pickle.dump(elmo_train_new,pickle_out)

       pickle_out.close()




       #save elmo_test_new

       pickle_out= open("elmo_test_03032019.pickle","wb")

       pickle.dump(elmo_test_new,pickle_out)

       pickle_out.close()

```

使用以下代码将其重新加载：

```
#load elmo_train_new

```

```
        pickle_in= open("elmo_train_03032019.pickle", "rb")

        elmo_train_new= pickle.load(pickle_in)









        #load elmo_train_new

        pickle_in= open("elmo_test_03032019.pickle", "rb")

        elmo_test_new= pickle.load(pickle_in)

```

8．建模与评估

让我们用ELMo建立NLP模型！

我们将使用训练数据集的ELMO向量来构建分类模型。然后，将使用该模型对测试集进行预测。但在所有这些之前，将elmo_train_new分为训练和验证集，以便在测试阶段之前评估模型。

```
fromsklearn.model_selection import train_test_split









        xtrain,xvalid, ytrain, yvalid = train_test_split(elmo_train_new,

                                                         train['label'],

                                                         random_state=42,

                                                         test_size=0.2)

```

由于目标是设定基线分数，我们将使用ELMo向量作为特征建立一个简单的逻辑回归模型：

```
fromsklearn.linear_model import LogisticRegression

        fromsklearn.metrics import f1_score



        lreg= LogisticRegression()

        lreg.fit(xtrain,ytrain)

```

是时候进行预测了！首先，在验证集上：

```
preds_valid = lreg.predict(xvalid)
```

因为F1评分标准是比赛的官方评价标准，所以我们用其来评价模型。

```
f1_score(yvalid, preds_valid)
```

输出: 0.789976

验证集的F1分数令人非常印象深刻。现在，让我们继续对测试集进行预测：

```
# make predictions on test set

preds_test = lreg.predict(elmo_test_new)
```

准备提交文件，将其上传到竞赛页面：

```
#prepare submission dataframe

        sub= pd.DataFrame({'id':test['id'], 'label':preds_test})



        #write predictions to a CSV file

        sub.to_csv("sub_lreg.csv",index=False)

```

这些预测使我们在公开排行榜上的得分为0.875672。坦率地说，这相当令人印象深刻，因为我们只做了最基本的文本预处理，并使用了一个非常简单的模型。

我们还能用ELMo做什么?

我们只是亲眼看到ELMo对文本分类是多么有效。如果再加上一个更复杂的模型，它肯定会提供更好的性能。ELMo的应用不仅限于文本分类任务。还可以将其用于文本数据的矢量化处理中。

利用ELMo还可完成更多NLP任务，如下所示：

- 机器翻译
- 语言建模
- 文本摘要
- 命名实体识别
- 问答系统

留言 点赞 发个朋友圈

我们一起分享AI学习与发展的干货