

转 从Word Embedding到Bert模型—自然语言处理中的预训练技术发展史

2018年12月22日 09:28:59 皮的开心 阅读数：144 标签： NLP ELMO Bert 更多

2018年作为 NLP的转折年，真的出现了很多优秀的模型，发现了大佬的一篇文章，对word embedding、ELMO、GPT、BERT之间的发展和联系进行了系统的对比，读完以后大彻大悟，学到了很多，赶紧关注粉一波。转载自张俊林大佬的微信文章，大家可以看一下大佬的博客，很多干货。<https://blog.csdn.net/malefactor>

Bert最近很火，应该是最近最火爆的AI进展，网上的评价很高，那么Bert值得这么高的评价吗？我个人判断是值得。那为什么会有这么高的评价呢？是因为它有重大的理论或者模型创新吗？其实并没有，从模型创新角度看一般，创新不算大。但是架不住效果太好了，基本刷新了很多NLP的任务的最好性能，有些任务还被刷爆了，这个才是关键。另外一点是Bert具备广泛的通用性，就是说绝大部分NLP任务都可以采用类似的两阶段模式直接去提升效果，这个第二关键。客观的说，把Bert当做最近两年NLP重大进展的集大成者更符合事实。

本文的主题是自然语言处理中的预训练过程，会大致说下NLP中的预训练技术是一步一步如何发展到Bert模型的，从中可以很自然地看到Bert的思路是如何逐渐形成的，Bert的历史沿革是什么，继承了什么，创新了什么，为什么效果那么好，主要原因是什么，以及为何说模型创新不算太大，为何说Bert是近年来NLP重大进展的集大成者。我们一步一步来讲，而串起来这个故事的脉络就是自然语言的预训练过程，但是落脚点还是在Bert身上。要讲自然语言的预训练，得先从图像领域的预训练说起。

图像领域的预训练

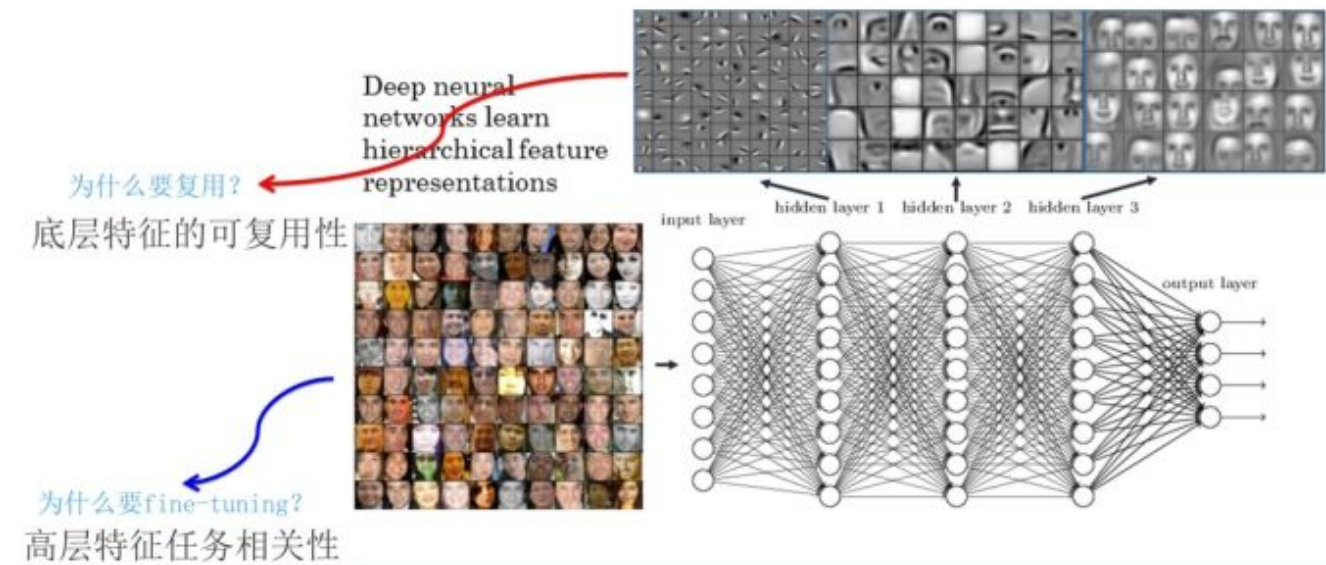
自从深度学习火起来后，预训练过程就是做图像或者视频领域的一种比较常规的做法，有比较长的历史了，而且这种做法很有效，能明显促进应用的效果。

那么图像领域怎么做预训练呢，上图展示了这个过程，我们设计好网络结构以后，对于图像来说一般是CNN的多层叠加网络结构，可以先用某个训练集比如训练集A或者训练集B对这个网络进行预先训练，在A任务上或者B任务上学会网络参数，然后存起来以备后用。假设我们面临第三个任务C，网络结构采取相同的网络结构，在比较浅的几层CNN结构，网络参数初始化的时候可以加载A任务或者B任务学习好的参数，其它CNN高层参数仍然随机初始化。之后我们用C任务的训练数据来训练网络，此时有两种做法，一种是浅层加载的参数在训练C任务过程中不动，这种方法被称为“Frozen”;另外一种一种是底层网络参数尽管被初始化了，在C任务训练过程中仍然随着训练的进程不断改变，这种一般叫“Fine-Tuning”，顾名思义，就是更好地把参数进行调整使得更适应当前的C任务。一般图像或者视频领域要做预训练一般都这么做。

这么做有几个好处，首先，如果手头任务C的训练集合数据量较少的话，现阶段的好用的CNN比如Resnet/Densenet/Inception等网络结构层数很深，几百万上千万参数量算起步价，上亿参数的也很常见，训练数据少很难很好地训练这么复杂的网络，但是如果其中大量参数通过大的训练集合比如ImageNet预先训练好直接拿来初始化大部分网络结构参数，然后再用C任务手头比较可怜的数据量上Fine-tuning过程去调整参数让它们更适合解决C任务，那事情就好办多了。这样原先训练不了的任务就能解决了，即使手头任务训练数据也不少，加个预训练过程也能极大加快任务训练的收敛速度，所以这种预训练方式是老少皆宜的解决方案，另外疗效又好，所以在做图像处理领域很快就流行开来。

那么新的问题来了，为什么这种预训练的思路是可行的？

预训练在图像领域的应用



目前我们已经知道，对于层级的CNN结构来说，不同层级的神经元学习到了不同类型的图像特征，由底向上特征形成层级结构，如上图所示，如果我们手头是个人脸

上抽取出的特征越与手头任务相关。正因为此，所以预训练好的网络参数，尤其是底层的网络参数抽取特征跟具体任务越无关，越具备任务的通用性，所以这是：一般用底层预训练好的参数初始化新任务网络参数的原因。而高层特征跟任务关联较大，实际可以不用使用，或者采用Fine-tuning用新数据集清洗掉高层无关的特征：器。

一般我们喜欢用ImageNet来做网络的预训练，主要有两点，一方面ImageNet是图像领域里有超多事先标注好训练数据的数据集合，分量足是个很大的优势，量越大出的参数越靠谱；另外一方面因为ImageNet有1000类，类别多，算是通用的图像数据，跟领域没太大关系，所以通用性好，预训练完后哪哪都能用，是个万金油。分……的万金油当然老少通吃，人人喜爱。

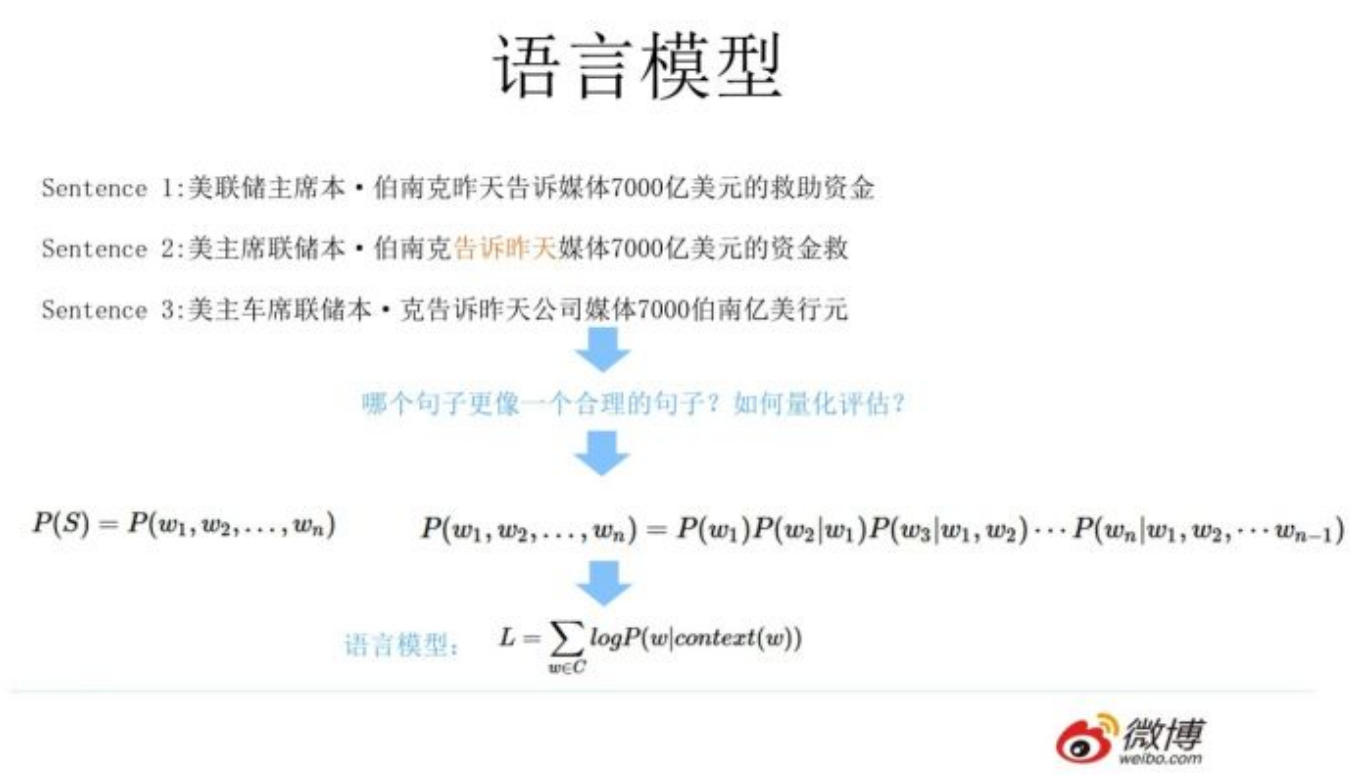
听完上述话，如果你是具备研究素质的人，也就是说具备好奇心，你一定会问下面这个问题：“既然图像领域预训练这么好用，那干嘛自然语言处理不做这个事情呢？是搞NLP的人比搞CV的傻啊？就算你傻，你看见人家这么做，有样学样不就行了吗？这不就是创新吗，也许能成，万一成了，你看，你的成功来得就是这么突然!”

嗯，好问题，其实搞NLP的人一点都不比你傻，早就有人尝试过了，不过总体而言不太成功而已。听说过word embedding吗？2003年出品，陈年技术，馥郁芳香。word embedding其实就是NLP里的早期预训练技术。当然也不能说word embedding不成功，一般加到下游任务里，都能有1到2个点的性能提升，只是没有那么耀眼的成功而已。

没听过？那下面就把这段陈年老账讲给你听听。

## Word Embedding考古史

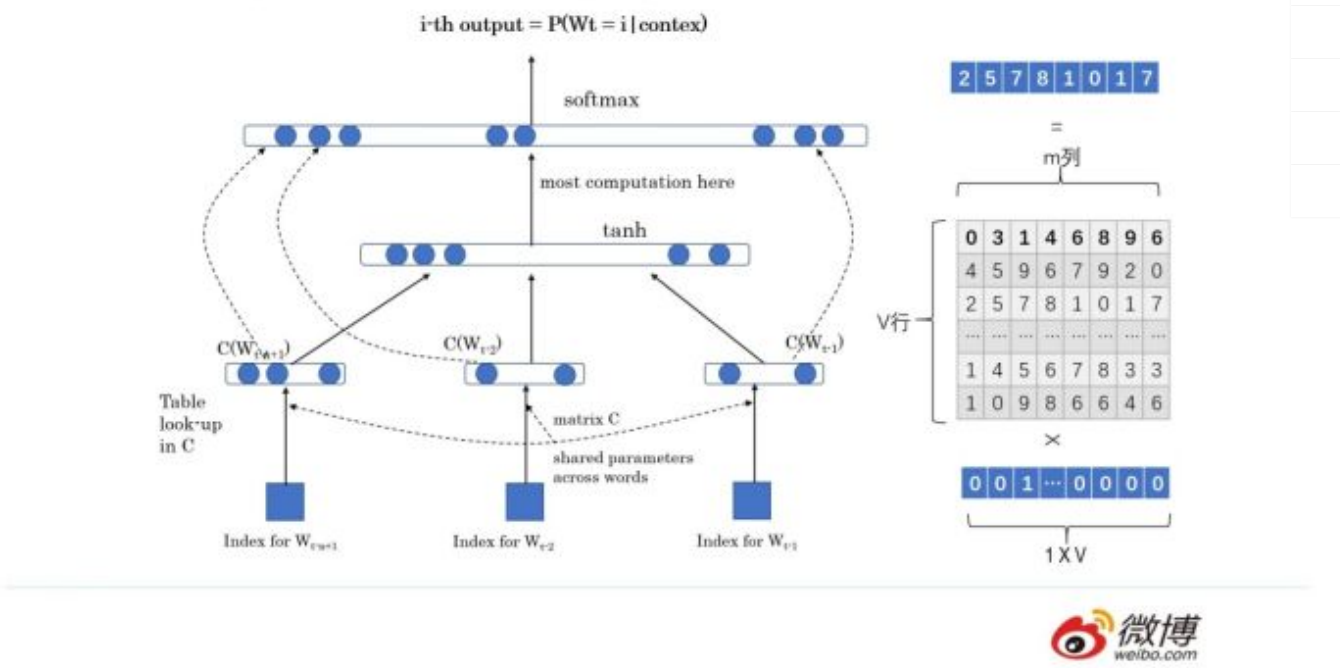
这块大致讲讲Word Embedding的故事，很粗略，因为网上关于这个技术讲的文章太多了，汗牛冲动，我不属牛，此刻更没有流汗，所以其实丝毫没有想讲Word Embedding的冲动和激情，但是要说预训练又得从这开始，那就粗略地讲讲，主要是引出后面更精彩的部分。在说Word Embedding之前，先更粗略地说下语言模型，因为一般NLP里面做预训练一般的选择是用语言模型任务来做。



什么是语言模型？其实看上面这张PPT上扣下来的图就明白了，为了能够量化地衡量哪个句子更像一句人话，可以设计如上图所示函数，核心函数P的思想是根据句子里面前面的一系列前导单词预测后面跟哪个单词的概率大小（理论上除了上文之外，也可以引入单词的下文联合起来预测单词出现概率）。句子里面每个单词都有个根据上文预测自己的过程，把所有这些单词的产生概率乘起来，数值越大代表这越像一句人话。语言模型压下暂且不表，我隐约预感到我这么讲你可能还是不太会明白，但是大概这个意思，不懂的可以去网上找，资料多得一样地汗牛冲动。

假设现在让你设计一个神经网络结构，去做这个语言模型的任务，就是说给你很多语料做这个事情，训练好一个神经网络，训练好之后，以后输入一句话的前面几个单词，要求这个网络输出后面紧跟的单词应该是哪个，你会怎么做？

# 神经网络语言模型（NNLM）



你可以像上图这么设计这个网络结构，这其实就是大名鼎鼎的中文人称“神经网络语言模型”，英文小名NNLM的网络结构，用来做语言模型。这个工作有年头了，是个陈年老工作，是Bengio在2003年发表在JMLR上的论文。它生于2003，火于2013，以后是否会不朽暂且不知，但是不幸的是出生后应该没有引起太大反响，沉寂十年终于时来运转沉冤得雪，在2013年又被NLP考古工作者从海底湿淋淋地捞出来了祭入神殿。为什么会发生这种技术奇遇记？你要想想2013年是什么年头，是深度学习开始渗透NLP领域的光辉时刻，万里长征第一步，而NNLM可以算是南昌起义第一枪。在深度学习火起来之前，极少有人用神经网络做NLP问题，如果你10年前坚持用神经网络做NLP，估计别人会认为你这人神经有问题。所谓红尘滚滚，谁也挡不住历史发展趋势的车轮，这就是个很好的例子。

上面是闲话，闲言碎语不要讲，我们回来讲一讲NNLM的思路。先说训练过程，现在看其实很简单，见过RNN、LSTM、CNN后的你们回头再看这个网络甚至显得有些简陋。学习任务是输入某个句中单词  $W_t = \text{“Bert”}$  前面句子的t-1个单词，要求网络正确预测单词Bert，即最大化：

$$P(W_t = \text{“Bert”} | W_1, W_2, \dots, W_{t-1}; \theta)$$

前面任意单词  $W_i$  用Onehot编码（比如：0001000）作为原始单词输入，之后乘以矩阵Q后获得向量  $C(W_i)$ ，每个单词的  $C(W_i)$  拼接，上接隐层，然后接softmax去预测后面应该后接哪个单词。这个  $C(W_i)$  是什么？这其实就是单词对应的Word Embedding值，那个矩阵Q包含V行，V代表词典大小，每一行内容代表对应单词的Word embedding值。只不过Q的内容也是网络参数，需要学习获得，训练刚开始用随机值初始化矩阵Q，当这个网络训练好之后，矩阵Q的内容被正确赋值，每一行代表一个单词对应的Word embedding值。所以你看，通过这个网络学习语言模型任务，这个网络不仅自己能够根据上文预测后接单词是什么，同时获得一个副产品，就是那个矩阵Q，这就是单词的Word Embedding是被如何学会的。

2013年最火的用语言模型做Word Embedding的工具是Word2Vec，后来又出了Glove，Word2Vec是怎么工作的呢？看下图。

Word2Vec的网络结构其实和NNLM是基本类似的，只是这个图长得清晰度差了点，看上去不像，其实它们是亲兄弟。不过这里需要指出：尽管网络结构相近，而且也是做语言模型任务，但是其训练方法不太一样。Word2Vec有两种训练方法，一种叫CBOW，核心思想是从一个句子里面把一个词抠掉，用这个词的上文和下文去预测被抠掉的这个词；第二种叫做Skip-gram，和CBOW正好反过来，输入某个单词，要求网络预测它的上下文单词。而你回头看看，NNLM是怎么训练的？是输入一个单词的上文，去预测这个单词。这是有显著差异的。为什么Word2Vec这么处理？原因很简单，因为Word2Vec和NNLM不一样，NNLM的主要任务是要学习一个解决语言模型任务的网络结构，语言模型就是要看到上文预测下文，而word embedding只是无心插柳的一个副产品。但是Word2Vec目标不一样，它单纯就是要word embedding的，这是主产品，所以它完全可以随性地这么去训练网络。

为什么要讲Word2Vec呢？这里主要是要引出CBOW的训练方法，BERT其实跟它有关系，后面会讲它们之间是如何的关系，当然它们的关系BERT作者没说，是我猜的，至于我猜的对不对，后面你看后自己判断。

使用Word2Vec或者Glove，通过做语言模型任务，就可以获得每个单词的Word Embedding，那么这种方法的效果如何呢？上图给了网上找的几个例子，可以看出有些例子效果还是很不错的，一个单词表达成Word Embedding后，很容易找出语义相近的其它词汇。

我们的主题是预训练，那么问题是Word Embedding这种做法能算是预训练吗？这其实就是标准的预训练过程。要理解这一点要看看学会Word Embedding后下游任务是怎么用它的。

假设如上图所示，我们有个NLP的下游任务，比如QA，就是问答问题，所谓问答问题，指的是给定一个问题X，给定另外一个句子Y，要判断句子Y是否是问题X的正确答案。问答问题假设设计的网络结构如上图所示，这里不展开讲了，懂得自然懂，不懂的也没关系，因为这点对于本文主旨来说不关键，关键是网络如何使用训练好的Word Embedding的。它的使用方法其实和前面讲的NNLM是一样的，句子中每个单词以Onehot形式作为输入，然后乘以学好的Word Embedding矩阵Q，就直接取出单词对应的Word Embedding了。这乍看上去好像是查表操作，不像是预训练的做法是吧？其实不然，那个Word Embedding矩阵Q其实就是网络Onehot层到embedding层映射的网络参数矩阵。所以你看到了，使用Word Embedding等价于什么？等价于把Onehot层到embedding层的网络用预训练好的参数矩阵Q初始化了。这跟前面讲的图像领域预训练过程其实是一样的，区别于图像领域预训练只能初始化部分网络参数，语言模型的参数就不能丢掉了，下游NLP任务使用训练好的Word Embedding的时候也常需要用到预训练好的参数矩阵Q。



像有两种做法，一种是Frozen，就是Word Embedding那层网络参数固定不动；另外一种Fine-Tuning，就是Word Embedding这层参数使用新的训练集合训练也跟着训练过程更新掉。

上面这种做法就是18年之前NLP领域里面采用预训练的典型做法，之前说过，Word Embedding其实对于很多下游NLP任务是有帮助的，只是帮助没有大到闪瞎眼睛的围观群众的双眼而已。那么新问题来了，为什么这样训练及使用Word Embedding的效果没有期待中那么好呢？答案很简单，因为Word Embedding有问题噢。这个问题比较弱智的答案，关键是Word Embedding存在什么问题？这其实是个好问题。

这片在Word Embedding头上笼罩了好几年的乌云是什么？是多义词问题。我们知道，多义词是自然语言中经常出现的现象，也是语言灵活性和高效性的一种体现。这对Word Embedding来说有什么负面影响？如上图所示，比如多义词Bank，有两个常用含义，但是Word Embedding在对bank这个单词进行编码的时候，是区分不开两个含义的，因为它们尽管上下文环境中出现的单词不同，但是在用语言模型训练的时候，不论什么上下文的句子经过word2vec，都是预测相同的单词bank，而同一个bank的是同一行的参数空间，这导致两种不同的上下文信息都会编码到相同的word embedding空间里去。所以word embedding无法区分多义词的不同语义，这就是它的比较严重的问题。

你可能觉得自己很聪明，说这可以解决啊，确实也有很多研究人员提出很多方法试图解决这个问题，但是从今天往回看，这些方法看上去都成本太高或者太繁琐了，有没有简单优美的解决方案呢？

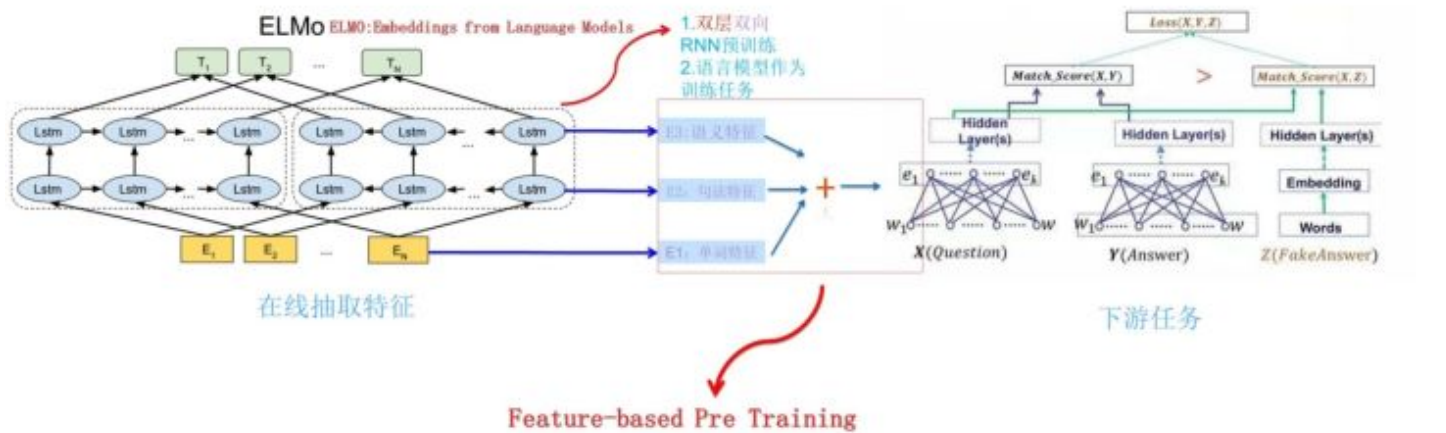
ELMO提供了一种简洁优雅的方案。

## 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。

ELMO采用了典型的两阶段过程，第一个阶段是利用语言模型进行预训练；第二个阶段是在做下游任务时，从预训练网络中提取对应单词的网络各层的Word Embedding作为新特征补充到下游任务中。上图展示的是其预训练过程，它的网络结构采用了双层双向LSTM，目前语言模型训练的任务目标是根据单词  $w_i$  的上下文去正确预测单词  $w_i$ ， $w_i$  之前的单词序列Context-before称为上文，之后的单词序列Context-after称为下文。图中左端的前向双层LSTM代表正方向编码器，输入的是从左到右顺序的除了预测单词外  $w_i$  的上文Context-before；右端的逆向双层LSTM代表反方向编码器，输入的是从右到左的逆序的句子下文Context-after；每个编码器的深度都是两层LSTM叠加。这个网络结构其实在NLP中是很常用的。使用这个网络结构利用大量语料做语言模型任务就能预先训练好这个网络，如果训练好这个网络后，输入一个新句子 *Snew*，句子中每个单词都能得到对应的三个Embedding:最底层是单词的Word Embedding，往上走是第一层双向LSTM中对应单词位置的Embedding，这层编码单词的句法信息更多一些；再往上走是第二层LSTM中对应单词位置的Embedding，这层编码单词的语义信息更多一些。也就是说，ELMO的预训练过程不仅仅学会单词的Word Embedding，还学会了一个双层双向的LSTM网络结构，而这两者后面都有用。

# ELMO：训练好之后如何使用？



上面介绍的是ELMO的第一阶段：预训练阶段。那么预训练好网络结构后，如何给下游任务使用呢？上图展示了下游任务的使用过程，比如我们的下游任务仍然是问题，此时对于问句X，我们可以先将句子X作为预训练好的ELMO网络的输入，这样句子X中每个单词在ELMO网络中都能获得对应的三个Embedding，之后给予这三个Embedding中的每一个Embedding一个权重 $a$ ，这个权重可以学习得来，根据各自权重累加和，将三个Embedding整合成一个。然后将整合后的这个Embedding放在自己任务的那个网络结构中对应单词的输入，以此作为补充的新特征给下游任务使用。对于上图所示下游任务QA中的回答句子Y来说也是如此处理。因为ELMO提供的是每个单词的特征形式，所以这一类预训练的方法被称为“Feature-based Pre-Training”。至于为何这么做能够达到区分多义词的效果，你可以想一想，其实特别明白原因。

上面这个图是TagLM采用类似ELMO的思路做命名实体识别任务的过程，其步骤基本如上述ELMO的思路，所以此处不展开说了。TagLM的论文发表在2017年的ACL上，作者就是AllenAI里做ELMO的那些人，所以可以将TagLM看做ELMO的一个前导工作。前几天这个PPT发出去后有人质疑说FastAI的在18年4月提出的ULFiT：弃传统Word Embedding引入新模式的开山之作，我深不以为然。首先TagLM出现的更早而且模式基本就是ELMO的思路；另外ULFiT使用的是三阶段模式，在通用型训练之后，加入了一个领域语言模型预训练过程，而且论文重点工作在这块，方法还相对比较繁杂，这并不是一个特别好的主意，因为领域语言模型的限制是它往往不可能特别大，精力放在这里不太合适，放在通用语言模型上感觉更合理；再者，尽管ULFiT实验做了6个任务，但是都集中在分类问题相对比较窄，不如ELMO验证的问题领域广，我觉得这就是因为第二步那个领域语言模型带来的限制。所以综合看，尽管ULFiT也是个不错的工作，但是重要性跟ELMO比至少还是要差一档，当然这是我个人看法。每个人的学术审美口味不同，我个人一直比较赞赏要么简洁有效体现问题本质要么思想特别游离现有框架脑洞开得异常大的工作，所以ULFiT我看论文的时候就感觉看着有点难受，觉得这工作没抓住重点而且特别麻烦，但是看ELMO论文感觉就赏心悦目，觉得思路特别清晰顺畅，看完暗暗点赞，心里说这样的文章获得NAACL2018最佳论文当之无愧，比ACL很多最佳论文也好得不是一点半点，这就是好工作带给一个有经验人士的一种在读论文时候就能产生的本能的感觉，也就是所谓的这道菜对上了食客的审美口味。

# ELMO：多义词问题解决了么？

(多义词)Play:  
1. 运动  
2. 音乐

Source	Nearest Neighbors
GloVe → play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM → Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
biLM → Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

Table 4: Nearest neighbors to “play” using GloVe and the context embeddings from a biLM.

不仅解决了，词性也能对应起来！



前面我们提到静态Word Embedding无法解决多义词的问题，那么ELMO引入上下文动态调整单词的embedding后多义词问题解决了吗？解决了，而且比我们期待的解决得还要好。上图给了个例子，对于Glove训练出的Word Embedding来说，多义词比如play，根据它的embedding找出的最接近的其它单词大多数集中在体育领域，这很明显是因为训练数据中包含play的句子中体育领域的数量明显占优导致；而使用ELMO，根据上下文动态调整后的embedding不仅能够找出对应的“演出”的相同语义的句子，而且还可以保证找出的句子中的play对应的词性也是相同的，这是超出期待之处。之所以会这样，是因为我们上面提到过，第一层LSTM编码了很多句法信息，这在这里起到了重要作用。

ELMO经过这般操作，效果如何呢？实验效果见上图，6个NLP任务中性能都有幅度不同的提升，最高的提升达到25%左右，而且这6个任务的覆盖范围比较广，包含句子语义关系判断，分类任务，阅读理解等多个领域，这说明其适用范围是非常广的，普适性强，这是一个非常好的优点。

那么站在现在这个时间节点看，ELMO有什么值得改进的缺点呢？首先，一个非常明显的缺点在特征抽取器选择方面，ELMO使用了LSTM而不是新贵Transformer，Transformer是谷歌在17年做机器翻译任务的“Attention is all you need”的论文中提出的，引起了相当大的反响，很多研究已经证明了Transformer提取特征的能力是要远强于LSTM的。如果ELMO采取Transformer作为特征提取器，那么估计Bert的反响远不如现在的这种火爆场面。另外一点，ELMO采取双向拼接这种融合特征的能力可能比Bert一体化的融合特征方式弱，但是，这只是一从道理推断产生的怀疑，目前并没有具体实验说明这一点。

我们如果把ELMO这种预训练方法和图像领域的预训练方法对比，发现两者模式看上去还是有很大差异的。除了以ELMO为代表的这种基于特征融合的预训练方法外，NLP里还有一种典型做法，这种做法和图像领域的方式就是看上去一致的了，一般将这种方法称为“基于Fine-tuning的模式”，而GPT就是这一模式的典型开创者。

## 从Word Embedding到GPT

2018/12/25                      从Word Embedding到Bert模型—自然语言处理中的预训练技术发展史 - asd2479745295的博客 - CSDN博客

提到过它的特征抽取能力要强于RNN，这个选择很明显是很明智的；其次，GPT的预训练虽然仍然是以语言模型作为目标任务，但是采用的是单向的语言模型，所

向'的含义是指：语言模型训练的任务目标是根据  $W_i$  单词的上下文去正确预测单词  $W_i$ ， $W_i$  之前的单词序列Context-before称为上文，之后的单词序列Context-after 为下文。ELMO在做语言模型预训练的时候，预测单词  $W_i$  同时使用了上文和下文，而GPT则只采用Context-before这个单词的上文来进行预测，而抛弃了下文。这个 见

在看不不是个太好的选择，原因很简单，它没有把单词的下文融合进来，这限制了其在更多应用场景的效果，比如阅读理解这种任务，在做任务的时候是可以允许同时 上文和下文一起做决策的。如果预训练时候不把单词的下文嵌入到Word Embedding中，是很吃亏的，白白丢掉了很多信息。

这里强行插入一段简单提下Transformer，尽管上面提到了，但是说的还不完整，补充两句。首先，Transformer是个叠加的"自注意力机制（Self Attention）"构成的 网络，是目前NLP里最强的特征提取器，注意力这个机制在此被发扬光大，从任务的配角不断抢戏，直到Transformer一跃成为踢开RNN和CNN传统特征提取器，荣升 大红大紫。你问了：什么是注意力机制？这里再插个广告，对注意力不了解的可以参考鄙人16年出品17年修正的下文：“深度学习中的注意力模型”，补充下相关基

础，如果不了解注意力机制你肯定会落后时代的发展。而介绍Transformer比较好的文章可以参考以下两篇文章：一个是Jay Alammar可视化地介绍Transformer的博客 [Illustrated Transformer](#)，非常容易理解整个机制，建议先从这里看起；然后可以参考哈佛大学NLP研究组写的“The Annotated Transformer.”，代码原理双管齐下，非常 清楚。我相信上面两篇文章足以让你了解Transformer了，所以这里不展开介绍。

其次，我的判断是Transformer在未来会逐渐替代掉RNN成为主流的NLP工具，RNN一直受困于其并行计算能力，这是因为它本身结构的序列性依赖导致的，尽管很多人在 试图通过修正RNN结构来修正这一点，但是我不看好这种模式，因为给马车换轮胎不如把它升级到汽车，这个道理很好懂，更何况目前汽车的雏形已经出现了，干嘛还要 执着在换轮胎这个事情呢？是吧？再说CNN，CNN在NLP里一直没有形成主流，CNN的最大优点是易于做并行计算，所以速度快，但是在捕获NLP的序列关系尤其是长距

离特征方面天然有缺陷，不是做不到而是做不好，目前也有很多改进模型，但是特别成功的不多。综合各方面情况，很明显Transformer同时具备并行性好，又适合捕获长 距离特征，没有理由不在赛跑比赛中跑不过RNN和CNN。

好了，题外话结束，我们再回到主题，接着说GPT。上面讲的是GPT如何进行第一阶段的预训练，那么假设预训练好了网络模型，后面下游任务怎么用？它有自己的个 性，和ELMO的方式大有不同。

上图展示了GPT在第二阶段如何使用。首先，对于不同的下游任务来说，本来你可以任意设计自己的网络结构，现在不行了，你要向GPT的网络结构看齐，把任务的网 络结构改造成和GPT的网络结构是一样的。然后，在做下游任务的时候，利用第一步预训练好的参数初始化GPT的网络结构，这样通过预训练学到的语言学知识就被引入到

你手头的任务里来了，这是个非常好的事情。再次，你可以用手头的任务去训练这个网络，对网络参数进行Fine-tuning，使得这个网络更适合解决手头的问题。就是这 样。看到了么？这有没有让你想起最开始提到的图像领域如何做预训练的过程（请参考上图那句非常容易暴露年龄的歌词）？对，这跟那个模式是一模一样的。

这里引入了一个新问题：对于NLP各种花样的不同任务，怎么改造才能靠近GPT的网络结构呢？

GPT论文给了一个改造施工图如上，其实也很简单：对于分类问题，不用怎么动，加上一个起始和终结符号即可；对于句子关系判断问题，比如Entailment，两个句子中间 再加个分隔符即可；对文本相似性判断问题，把两个句子顺序颠倒下做出两个输入即可，这是为了告诉模型句子顺序不重要；对于多项选择问题，则多路输入，每一路把文

章和答案选项拼接作为输入即可。从上图可看出，这种改造还是很方便的，不同任务只需要在输入部分施工即可。

GPT的效果是非常令人惊艳的，在12个任务里，9个达到了最好的效果，有些任务性能提升非常明显。

那么站在现在的时间节点看，GPT有什么值得改进的地方呢？其实最主要的就是那个单向语言模型，如果改造成双向的语言模型任务估计也没有Bert太多事了。当然，即使 如此GPT也是非常非常好的一个工作，跟Bert比，其作者炒作能力亟待提升。

Bert的诞生

从GPT和ELMO及Word2Vec到Bert：新星的诞生



Bert采用和GPT完全相同的两阶段模型，首先是语言模型预训练；其次是使用Fine-Tuning模式解决下游任务。和GPT的最主要不同在于在预训练阶段采用了类似ELMO的双向语言模型，当然另外一点是语言模型的数据规模要比GPT大。所以这里Bert的预训练过程不必多讲了。

第二阶段，Fine-Tuning阶段，这个阶段的做法和GPT是一样的。当然，它也面临着下游任务网络结构改造的问题，在改造任务方面Bert和GPT有些不同，下面简单介绍一下。

在介绍Bert如何改造下游任务之前，先大致说下NLP的几类问题，说这个是为了强调Bert的普适性有多强。通常而言，绝大部分NLP问题可以归入上图所示的四类问题。第一类是序列标注，这是最典型的NLP任务，比如中文分词，词性标注，命名实体识别，语义角色标注等都可以归入这一类问题，它的特点是句子中每个单词要求模型给出一个分类类别。第二类是分类任务，比如我们常见的文本分类，情感计算等都可以归入这一类。它的特点是不管文章有多长，总体给出一个分类类别。第三类任务是句子关系判断，比如Entailment，QA，语义改写，自然语言推理等任务都是这个模式，它的特点是给定两个句子，模型判断出两个句子是否具备某种语义关系；第四类是生成式任务，比如机器翻译，文本摘要，写诗造句，看图说话等都属于这一类。它的特点是输入文本内容后，需要自主生成另外一段文字。

对于种类如此繁多而且各具特点的下游NLP任务，Bert如何改造输入输出部分使得大部分NLP任务都可以使用Bert预训练好的模型参数呢？上图给出示例，对于句子关系类任务，很简单，和GPT类似，加上一个起始和终结符号，句子之间加个分隔符即可。对于输出来说，把第一个起始符号对应的Transformer最后一层位置上面串接一个softmax分类层即可。对于分类问题，与GPT一样，只需要增加起始和终结符号，输出部分和句子关系判断任务类似改造；对于序列标注问题，输入部分和单句分类是一样的，只需要输出部分Transformer最后一层每个单词对应位置都进行分类即可。从这里可以看出，上面列出的NLP四大任务里面，除了生成类任务外，Bert其它都覆盖到了，而且改造起来很简单直观。尽管Bert论文没有提，但是稍微动动脑子就可以想到，其实对于机器翻译或者文本摘要，聊天机器人这种生成式任务，同样可以稍作改造即可引入Bert的预训练成果。只需要附着在S2S结构上，encoder部分是个深度Transformer结构，decoder部分也是个深度Transformer结构。根据任务选择不同的预训练数据初始化encoder和decoder即可。这是相当直观的一种改造方法。当然，也可以更简单一点，比如直接在单个Transformer结构上加装隐层产生输出也是可以的。不论如何，从这里可以看出，NLP四大类任务都可以比较方便地改造成Bert能够接受的方式。这其实是Bert的非常大的优点，这意味着它几乎可以做任何NLP的下游任务，具备普适性，这是很强的。

Bert采用这种两阶段方式解决各种NLP任务效果如何？在11个各种类型的NLP任务中达到目前最好的效果，某些任务性能有极大的提升。一个新模型好不好，效果才是王道。

到这里我们可以再梳理下几个模型之间的演进关系。从上图可见，Bert其实和ELMO及GPT存在千丝万缕的关系，比如如果我们把GPT预训练阶段换成双向语言模型，那么就得到了Bert；而如果我们把ELMO的特征抽取器换成Transformer，那么我们会得到Bert。所以你可以看出：Bert最关键两点，一点是特征抽取器采用Transformer；第二点是预训练的时候采用双向语言模型。

那么新问题来了：对于Transformer来说，怎么才能在这个结构上做双向语言模型任务呢？乍一看上去好像不太好搞。我觉得吧，其实有一种很直观的思路，怎么办？看看ELMO的网络结构图，只需要把两个LSTM替换成两个Transformer，一个负责正向，一个负责反向特征提取，其实应该就可以。当然这是我自己的改造，Bert没这么做。那么Bert是怎么做的呢？我们前面不是提过Word2Vec吗？我前面肯定不是漫无目的地提到它，提它是为了在这里引出那个CBOW训练方法，所谓写作时候埋伏笔的“草蛇灰线，伏脉千里”，大概就是这个意思吧？前面提到了CBOW方法，它的核心思想是：在做语言模型任务的时候，我把要预测的单词抠掉，然后根据它的上文Context-Before和下文Context-after去预测单词。其实Bert怎么做的？Bert就是这么做的。从这里可以看到方法间的继承关系。当然Bert作者没提Word2Vec及CBOW方法，这是我的判断，Bert作者说是受到完形填空任务的启发，这也很有可能，但是我觉得他们要是没想到过CBOW估计是不太可能的。

从这里可以看出，在文章开始我说过Bert在模型方面其实没有太大创新，更像一个最近几年NLP重要技术的集大成者，原因在于此，当然我不确定你怎么看，是否认同这种看法，而且我也不关心你怎么看。其实Bert本身的效果好和普适性强才是最大的亮点。

那么Bert本身在模型和方法角度有什么创新呢？就是论文中指出的Masked 语言模型和Next Sentence Prediction。而Masked语言模型上面讲了，本质思想其实是CBOW，但是细节方面有改进。

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双向语言模型的具体做法。

至于说“Next Sentence Prediction”，指的是做语言模型预训练的时候，分两种情况选择两个句子，一种是选择语料中真正顺序相连的两个句子；另外一种是从语料库中抛色子，随机选择一个拼到第一个句子后面。我们要求模型除了做上述的Masked语言模型任务外，附带再做个句子关系预测，判断第二个句子是不是真的是第一个句子的后续句子。之所以这么做，是考虑到很多NLP任务是句子关系判断任务，单词预测粒度的训练到不了句子关系这个层级，增加这个任务有助于下游句子关系判断任务。所以可以看到，它的预训练是个多任务过程。这也是Bert的一个创新。

上面这个图给出了一个我们此前利用微博数据和开源的Bert做预训练时随机抽出的一个中文训练实例，从中可以体会下上面讲的masked语言模型和下句预测任务。训练数据就长这种样子。

顺带讲解下Bert的输入部分，也算是有些特色。它的输入部分是线性序列，两个句子通过分隔符分割，最前面和最后增加两个标识符号。每个单词有三个embedding:位置信息embedding，这是因为NLP中单词顺序是很重要的特征，需要在这里对位置信息进行编码；单词embedding,这个就是我们之前一直提到的单词embedding；第三个是句子embedding，因为前面提到训练数据都是由两个句子构成的，那么每个句子有个句子整体的embedding项对应给每个单词。把单词对应的三个embedding叠加，就形成了Bert的输入。

至于Bert在预训练的输出部分如何组织，可以参考上图的注释。

我们说过Bert效果特别好，那么到底是什么因素起作用呢？如上图所示，对比试验可以证明，跟GPT相比，双向语言模型起到了最主要的作用，对于那些需要看到下文的任务来说尤其如此。而预测下个句子来说对整体性能来说影响不算太大，跟具体任务关联度比较高。

最后，我讲讲我对Bert的评价和看法，我觉得Bert是NLP里程碑式的工作，对于后面NLP的研究和工业应用会产生长久的影响，这点毫无疑问。但是从上文介绍也可以看出，从模型或者方法角度看，Bert借鉴了ELMO，GPT及CBOW，主要提出了Masked 语言模型及Next Sentence Prediction，但是这里Next Sentence Prediction基本不影响大局，而Masked LM明显借鉴了CBOW的思想。所以说Bert的模型没什么大的创新，更像最近几年NLP重要进展的集大成者，这点如果你看懂了上文估计也没有太大异议，如果你有大的异议，杠精这个大帽子我随时准备戴给你。如果归纳一下这些进展就是：首先是两阶段模型，第一阶段双向语言模型预训练，这里注意要用双向而不是单向，第二阶段采用具体任务Fine-tuning或者做特征集成；第二是特征抽取要用Transformer作为特征提取器而不是RNN或者CNN；第三，双向语言模型可以采取CBOW的方法去做（当然我觉得这个是个细节问题，不算太关键，前两个因素比较关键）。Bert最大的亮点在于效果好及普适性强，几乎所有NLP任务都可以套用Bert这种两阶段思路，而且效果应该会有明显提升。可以预见的是，未来一段时间在NLP应用领域，Transformer将占据主导地位，而且这种两阶段预训练方法也会主导各种应用。



2018/12/25

从Word Embedding到Bert模型－自然语言处理中的预训练技术发展史 - asd2479745295的博客 - CSDN博客

另外，我们应该清楚预训练这个过程本质上是在做什么事情，本质上预训练是通过设计好一个网络结构来做语言模型任务，然后把大量甚至是无穷尽的无标注的语料文本利用起来，预训练任务把大量语言学知识抽取出来编码到网络结构中，当手头任务带有标注信息的数据有限时，这些先验的语言学特征当然会对手头任务有积极的补充作用，因为当数据有限的时候，很多语言学现象是覆盖不到的，泛化能力就弱，集成尽量通用的语言学知识自然会加强模型的泛化能力。如何引入先验的语言学知识其实一直是NLP尤其是深度学习场景下的NLP的主要目标之一，不过一直没有太好的解决办法，而ELMO/GPT/Bert的这种两阶段模式看起来无疑是解决这个问题自然的方法，这也是这些方法的主要价值所在。

对于当前NLP的发展方向，我个人觉得有两点非常重要，一个是需要更强的特征抽取器，目前看Transformer会逐渐担当大任，但是肯定还是不够强的，需要发展更强的特征抽取器；第二个就是如何优雅地引入大量无监督数据中包含的语言学知识，注意我这里强调地是优雅，而不是引入，此前相当多的工作试图做各种语言学知识的强行引入，但是很多方法看着让人牙疼，就是我说的不优雅。目前看预训练这种两阶段方法还是很有效的，也非常简洁，当然后面肯定还会有更好的模型出现。

完了，这就是自然语言模型预训练的发展史。

 想对作者说点什么

下载

预训练在自然语言处理的发展，从Word Embedding到BERT模型

11-23

预训练在自然语言处理的发展，从Word Embedding到BERT模型，45页PPT

自然语言处理(NLP)之Word Embedding

2755

最近做完UNIT一个小项目后,结合基于知识图谱的问答,对NLP/NLU方向产生了比较... 来自: L\_R\_H000的博客

谷歌开源 BERT：最先进的 NLP 预训练技术，支持中文和更多语言

139

谷歌本周开源了一种 NLP 预训练新技术Bidirectional Encoder Representations from ... 来自: TensorFlowers...

[自然语言处理] 预训练词向量百度云下载 [Word2Vec, Glove, FastText]

3338

English Pre-trained word embeddings Google's word2vec embedding: 外网地址: [... 来自: LeYOUNGER的...

BERT 现已开源：最先进的 NLP 预训练技术，支持中文和更多语言

832

... 来自: 谷歌开发者

Word Embedding模型: 词、短语及它们的组合的分布式表示

2987

摘要 最近引入的连续Skip-gram模型是学习可以高质量分布式向量表示的有效方... 来自: qq\_20599123的...

NLP的游戏规则从此改写？从word2vec, ELMo到BERT

585

NLP的游戏规则从此改写？从word2vec, ELMo到BERT 2018-10-24 10:20| 发布者: ... 来自: chunyun0716的...

在Keras模型中使用预训练的词向量

911

转自: wwt.me/2017/08/21/pre-trained-embedding-keras/ 代码下载: https://github.... 来自: lrt366的博客

比赛必备 | 省力搞定三款词向量训练 + OOV词向量问题的可性方案

669

本篇为资源汇总，一些NLP的比赛在抽取文本特征的时候会使用非常多的方式。传... 来自: 素质云笔记/Rec...

文章热词

体感技术 体感技术视频教程 体感技术课程 体感技术学习 体感技术课程

相关热词

c++调用python训练的模型 android 发展史 手机 c# 中的word类库 c++中arma模型应用实例 c++中word用法 人工智能

词嵌入来龙去脉 word embedding、word2vec

9397

0词嵌入来龙去脉 NLP的核心关键语言表示Representation NLP词的表达方法类型 1... 来自: 博客堂



张俊林博客

119篇文章

排名:2000+

关注



悟乙己

302篇文章

排名:444

关注



TangowL

112篇文章

排名:9000+

关注



AI科技大本营

1611篇文章

排名:283

关注

谷歌BERT预训练模型开源第一手解读，NLPers的福利送上

453

BERT(Bidirectional Encoder Representations from Transformers), 基于自注意力机... 来自: weixin\_3952261...

pytorch加载已训练好的word-embedding

5007

十一的时候已将pytorch的tutorial看过了，但是并没有用pytorch做什么项目，一直以... 来自: Deep Learning ...

文献笔记|【文本表征】BERT

513

论文信息： 论文题目：BERT: Pre-training of Deep Bidirectional Transformers for La... 来自: 人工智能--王小...

怎样用预训练模型搞定深度学习？

582

本文主要介绍如何应用预训练模型来对自然语言处理任务进行建模，并给出一些... 来自: 人工智能--王小...



<b>自然语言处理中的语言模型预训练方法</b> ...	<div><div>👁 490</div><div>来自: <a href="#">Paper weekly</a></div></div>	0
<b>tensorflow：使用预训练词向量</b> 目前使用深度网络进行文本任务模型训练时，第一步应该是将文本转为词向量进行处...	<div><div>👁 1.9万</div><div>来自: <a href="#">xgli的博客</a></div></div>	
<b>韩国小哥哥用Pytorch实现谷歌最强NLP预训练模型BERT   代码</b> ...	<div><div>👁 589</div><div>来自: <a href="#">量子位</a></div></div>	
<b>BERT介绍</b> 这篇介绍以下最近大热的[BERT](https://arxiv.org/pdf/1810.04805.pdf)，它在11个NL...	<div><div>👁 7803</div><div>来自: <a href="#">triplemeng的博客</a></div></div>	
<b>流水账   Elmo词向量中文训练过程杂记</b> 1 elmo是什么？ 参考：《文本嵌入的经典模型与最新进展》 人们已经提出了大量...	<div><div>👁 6726</div><div>来自: <a href="#">素质云笔记/Rec...</a></div></div>	
<b>tensorflow使用预训练词向量</b> 目前使用深度网络进行文本任务模型训练时，第一步应该是将文本转为词向量进行处...	<div><div>👁 1726</div><div>来自: <a href="#">fkytly的专栏</a></div></div>	
<b>Keras中加载预训练的词向量</b> 经常需要加载预训练的词向量，有时候会脑抽忘记，所以记录下。vocab={} # 词汇...	<div><div>👁 711</div><div>来自: <a href="#">山不过来，我就...</a></div></div>	
<b>用深度学习实现自然语言处理：word embedding，单词向量化</b> 前几年，腾讯新闻曾发出一片具有爆炸性的文章。并不是文章的内容有什么新奇之处...	<div><div>👁 218</div><div>来自: <a href="#">tyler_download...</a></div></div>	
<b>词向量技术   从word2vec到ELMo</b> ...	<div><div>👁 1602</div><div>来自: <a href="#">leadai的博客</a></div></div>	
<b>BERT模型</b> 1、背景 语言模型的与训练在自然语言处理的任务中有着重要的作用。 1.1 自然语言...	<div><div>👁 297</div><div>来自: <a href="#">笔记杂谈</a></div></div>	
<b>ELMo——一种新型的词嵌入方法</b> 目录 一、概要 二、双向语言模型 三、线性组合 一、概要 英语论文原文参考链接：...	<div><div>👁 3729</div><div>来自: <a href="#">cpluss的博客</a></div></div>	
<b>谷歌最强 NLP 模型 BERT 解读</b> 从elmo--&gt;ULMFIT--&gt;gpt--&gt;bert，大概是个流程 最近谷歌...	<div><div>👁 536</div><div>来自: <a href="#">luoyexuge的专栏</a></div></div>	
<b>谷歌BERT预训练源码解析（三）：训练过程</b> 目录前言源码解析主函数自定义模型遮蔽词预测下一句预测规范化数据集 前言 本部...	<div><div>👁 221</div><div>来自: <a href="#">weixin_3947074...</a></div></div>	
<b>从compositional到distributed，从无监督到有监督再到多任务学习 —— ...</b> 关于词向量，word embedding，研究相对较早，存在多种表征方式，目前 distribute...	<div><div>👁 1094</div><div>来自: <a href="#">TangowL</a></div></div>	
<b>还在为没有中文训练数据而头痛你的NLP任务吗？是时候看看这篇文章了~</b> 编者：王小草 时间：2017年7月18日 1 摘要 近期对于跨语言词嵌入的研究虽然已经...	<div><div>👁 461</div><div>来自: <a href="#">人工智能--王小...</a></div></div>	
<b>[NLP自然语言处理]谷歌BERT模型深度解析</b> BERT模型代码已经发布，可以在我的github: NLP-BERT--Python3.6-pytorch 中下载...	<div><div>👁 1.4万</div><div>来自: <a href="#">Machine Learni...</a></div></div>	
<b>pytorch中的embedding词向量的使用</b> Embedding 词嵌入在 pytorch 中非常简单，只需要调用 torch.nn.Embedding(m, n) ...	<div><div>👁 3271</div><div>来自: <a href="#">david0611的博客</a></div></div>	
<b>关于句子embedding的一些工作简介（一）</b> 关于句子embedding的一些工作简介（一） 对于NLP方面的工作来讲，毫无疑问词...	<div><div>👁 1060</div><div>来自: <a href="#">triplemeng的博客</a></div></div>	
<b>利用Word Embedding自动生成语义相近句子</b> 本文给出了使用Word Embedding来自自动生成语义相近句子的运行思路，并给出了机...	<div><div>👁 1.5万</div><div>来自: <a href="#">张俊林的博客</a></div></div>	
<b>使用预训练的word2vec词向量</b> 以谷歌开源google news (bin) 为例。下载地址：https://code.google.com/p/word2v...	<div><div>👁 7757</div><div>来自: <a href="#">码世界</a></div></div>	
<b>word embedding的模型与测试</b> 模型和测评	<div><div>👁 1348</div><div>来自: <a href="#">小丫头的博客</a></div></div>	

<div>使用<b>Word Embedding</b>构造简洁有效的文本摘要系统</div> <div>本文介绍了使用Word Embedding进行摘要提取的一种非常简洁的方法。</div>	<div>👁 2.8万</div> <div>来自: <a href="#">张俊林的博客</a></div>	0
<div><b>模型</b>微调fine-tuning</div> <div>因为上一篇博客采用的数据集太少，直接利用模型进行训练其预测能力较弱，故在此...</div>	<div>👁 132</div> <div>来自: <a href="#">王旭的博客</a></div>	
<div><b>Google</b>最强<b>模型BERT</b>出炉，NLP还有哪些值得期待的发展？</div> <div>AI技术年度盛会即将开启！11月8-9日，来自Google、Amazon、微软、Facebook、...</div>	<div>👁 329</div> <div>来自: <a href="#">AI科技大本营</a></div>	
<div>文本处理——fastText原理及实践（四）</div> <div>博文地址：<a href="https://zhuanlan.zhihu.com/p/32965521">https://zhuanlan.zhihu.com/p/32965521</a>fastText是Facebook于2016年开...</div>	<div>👁 1200</div> <div>来自: <a href="#">爱吃鱼的猫</a></div>	
<div><div>下载</div>深度学习<b>word embedding</b> 下载</div> <div>2--深度学习--word embedding 很宝贵的资料。超清晰ppt资源下载地址，适合深入学习机器学习和深度学习的人群。下载请慎重，10分。资源绝对宝贵，初学者请慎重下载。</div>	06-28	
<div>使用Keras和预训练的词向量训练新闻文本分类<b>模型</b></div> <div>from __future__ import print_function import os import sys import numpy as np from...</div>	<div>👁 755</div> <div>来自: <a href="#">大雄没有叮当猫...</a></div>	
<div>两行代码玩转 Google <b>BERT</b> 句向量词向量</div> <div>...</div>	<div>👁 624</div> <div>来自: <a href="#">Paper weekly</a></div>	
<div>词袋<b>模型</b>（BOW，bag of <b>words</b>）和词向量<b>模型</b>（<b>Word Embedding</b>）概...</div> <div>例句:Jane wants to go to Shenzhen.Bob wants to go to Shanghai.一、词袋模型 ...</div>	<div>👁 1611</div> <div>来自: <a href="#">随风、逆风的博客</a></div>	
<div>逐步视频讲解--用Tensorflow进行中文<b>自然语言处理</b>--情感分析</div> <div>本教程为原创，转载请注明教学视频地址， 视频教程链接：<a href="https://www.bilibili.com/v...">https://www.bilibili.com/v...</a></div>	<div>👁 409</div> <div>来自: <a href="#">AESPRESSO</a></div>	
<div><b>【数据应用技巧】NLP领域的预训练之风</b></div> <div>案例来源：<a href="#">@AI科技评论</a> <a href="#">@集智翻译组</a> <a href="#">@人工智能LeadAI</a> 案例地址：<a href="https://mp.we...">https://mp.we...</a></div>	<div>👁 276</div> <div>来自: <a href="#">数据案例笔记</a></div>	
<div><b>【python 走进NLP】NLP WordEmbedding的概念和实现</b></div> <div>背景 如何表示词语所包含的语义？苹果？水果？Iphone？ 苹果、梨子，这两个词相...</div>	<div>👁 3416</div> <div>来自: <a href="#">赖德发的博客</a></div>	
<div>使用中文wiki语料库训练word2vec</div> <div>1. 任务描述本文主要完成以下任务：训练字的embedding（unigram char embeddin...</div>	<div>👁 917</div> <div>来自: <a href="#">zhyongwei的博客</a></div>	
<div>词嵌入<b>模型</b>浅析——<b>Word2vec</b>与glove</div> <div>我们在进行自然语言处理（NLP）时，面临的首要问题就是怎么将自然语言输...</div>	<div>👁 166</div> <div>来自: <a href="#">Jay5311的博客</a></div>	
<div>webstorm 2018 激活破解方法大全</div> <div>webstorm 作为最近最火的前端开发工具,也确实对得起那个价格,但是秉着勤俭节约...</div>	<div>👁 699541</div> <div>来自: <a href="#">唐大帅的编程之路</a></div>	
<div>刚发布！Python2 月再夺语言榜首！一二线城市月薪增长迅猛！</div> <div>几年前Python在国内还只是一门小众语言，但从2011开始到现在，Python的百度搜...</div>	<div>👁 5344</div> <div>来自: <a href="#">CSDN学院</a></div>	
<div>基于XMLHttpRequest对象的ajax拦截</div> <div>导读 在web前端开发中，我们经常会与后端进行数据的交互，ajax即是其中的一...</div>	<div>👁 157</div> <div>来自: <a href="#">幽幽小春</a></div>	
<div>js常用的正则验证</div> <div>一、手机号码验证 （一）中国电信号段：133、149、153、173、177、180、181...</div>	<div>👁 1592</div> <div>来自: <a href="#">幽幽小春</a></div>	
<div>pyCharm最新2018激活码</div> <div>本教程对jetbrains全系列可用例：IDEA、WebStorm、phpstorm、clion等 因公司的...</div>	<div>👁 1281226</div> <div>来自: <a href="#">昌昌</a></div>	
<div>很黄很暴力的十个网站</div> <div>13岁的北京学生张某，在去年12月27日19时新闻联播一则关于净化网络视听的新闻...</div>	<div>👁 83915</div> <div>来自: <a href="#">Kinb_huangwei...</a></div>	
<div>微服务Springcloud超详细教程+实战（十）</div> <div>本人正在找深圳Java实习工作，求大佬带飞 QQ：1172796094 如在文档中遇到什么...</div>	<div>👁 18229</div> <div></div>	
<div>颈椎病怎么能治好，看我用Python对接</div> <div>盘点现代社会中年轻人常见病例，很显然"颈椎病"一定是排在第一的。年轻人长期伏...</div>	<div>👁 15169</div> <div>来自: <a href="#">john_dung的博客</a></div>	

<b>最新迅雷“应版权方要求，文件无法下载”的解决办法</b>	👁 235087	
迅雷下载有的电影电视剧的时候会出现：应版权方要求，文件无法下载，或者显示迅... 来自： <a href="#">徐奕的专栏</a>		0
<b>整理了10个干净、好用的BT、磁力链搜索网站给大家</b>	👁 116421	
现在越来越流行在线看视频了，但是对于我得收藏癖爱好者，还是希望可以比较好... 来自： <a href="#">YXAPP的技术...</a>		
<b>利用k-means算法对点云数据进行目标分割，提取其中的建筑物、房屋等</b>	👁 115	
原始点云数据在CloudCompare的显示如下： 点云原始数据以及提取出的房屋道路等... 来自： <a href="#">qq_39343904的...</a>		
<b>颈椎病的症状,Pyhton结合治疗仪让治疗与众不同</b>	👁 10994	
第1部分 Python内建对象 第1章 Python对象初探 1.1 Python内的对象 1.1.1 对象机制... 来自： <a href="#">john_dung的博客</a>		
<b>“戏”说设计模式——外观（门面）模式</b>	👁 1956	
首先请出今天的主人公——“黑旋风”李逵 李逵：“我是不是萌萌哒？” 一部《水浒传... 来自： <a href="#">青衣煮茶</a>		
<b>webgl第26课-在图形上贴图片</b>	👁 4252	
需要电子档书籍可以Q群：828202939 希望可以和大家一起学习、一起进步！！ 上... 来自： <a href="#">谷子的博客</a>		
<b>webgl第30课-用键盘来改变视点</b>	👁 4604	
需要电子档书籍可以Q群：828202939 希望可以和大家一起学习、一起进步！！ 所... 来自： <a href="#">谷子的博客</a>		
<b>Webstorm 最新激活码 多种破解方式(持续更新...)</b>	👁 92692	
方法：License server 注册 安装完成，打开Webstorm，在弹出的License Activation... 来自： <a href="#">老妖儿的博客</a>		
<b>微服务Springcloud超详细教程+实战（八）</b>	👁 21531	
如在文档中遇到什么问题请联系作者 QQ：1172796094 本人正在找深圳Java实习工...		
<b>安装和激活Office 2019</b>	👁 33736	
有条件请支持正版！相比费尽力气找一个可能不太安全的激活工具，直接买随时随地... 来自： <a href="#">过了即是客</a>		
<b>java缓冲区</b>	👁 15668	
1 缓冲区的分类 ByteBuffer CharBuffer ShortBuffer IntBuffer LongBuffer FloatBuffer ... 来自： <a href="#">weixin_4369414...</a>		
<b>php调用其他接口的post或者get或者delete或者put的方式</b>	👁 5264	
1.PHP cURL函数基本列表 curl_close() //关闭一个cURL会话 curl_copy_handle() //复...		
<b>Redis(内存高速缓存数据库)简介及安装</b>	👁 551	
一.Redis 1.什么是Redis Redis是REmote DIctionary Server（远程数据服务）的缩写... 来自： <a href="#">Tang5ge的博客</a>		
<b>Proxyee-down的下载与安装教程</b>	👁 149957	
Proxyee-down是monkeyWie在Github上的一个开源项目，向作者致敬。最新版的Pr... 来自： <a href="#">shadandaejian...</a>		
<b>webgl第27课-三维空间-不同位置的三角形</b>	👁 4509	
需要电子档书籍可以Q群：828202939 希望可以和大家一起学习、一起进步！！ 上... 来自： <a href="#">谷子的博客</a>		
<b>军事理论课答案（西安交大版）</b>	👁 1156262	
1.1 1 【单选题】我国陆地领土面积排名世界第几？（C） A、1 B、2 C、3 D、4 2 ... 来自： <a href="#">ling_wang的博客</a>		
<b>Kotlin详解：第三章，实战</b>	👁 1872	
一，搭建开发环境 1，安装Android Studio 3.0及以上版本 2，升级Gradle插件版本至... 来自： <a href="#">wang_yong_hui...</a>		
<b>webgl第28课-三维空间之加入旋转矩阵的三角形</b>	👁 4638	
需要电子档书籍可以Q群：828202939 希望可以和大家一起学习、一起进步！！ 上... 来自： <a href="#">谷子的博客</a>		
<b>vue-cli(vue脚手架)快速搭建前端项目——Vue系列博客一</b>	👁 705	
vue脚手架vue-cli搭建前端项目，并且路由分模块 来自： <a href="#">Tang5ge的博客</a>		
<b>微服务Springcloud超详细教程+实战（九）</b>	👁 17424	
如在文档中遇到什么问题请联系作者 QQ：1172796094 本人正在找深圳Java实习工...		
<b>webgl第24课-几何图形的装配和光栅化</b>	👁 4407	
需要电子档书籍可以Q群：828202939 希望可以和大家一起学习、一起进步！！ 纯... 来自： <a href="#">谷子的博客</a>		
<b>智慧树走进故宫章节测试答案智慧树走进故宫网课插件2018年智慧树走...</b>	👁 116975	
欢迎通过微信客户端搜索jitu2019或者扫描下方二维码关注廖图公众号获取智慧树刷... 来自： <a href="#">柠檬很酸的博客</a>		
<b>利用Keras搭建神经网络进行回归预测</b>	👁 70	

- vs2017安装和使用教程（详细）

VS2017如此强大，不仅仅是C语言，Python，ios，Android，Web，Node.js，Azur...

来自： qq\_36556893的...

225475
- 各大磁力种子搜索引擎对比

现在磁力种子搜索引擎质量参差不齐，现在就重点整理几个常用的种子搜索站，做个...

来自： lizhengnanhua...

449826
- 手把手教你如何安装Pycharm——靠谱的Pycharm安装详细教程

今天小编给大家分享如何在本机上下载和安装Pycharm，具体的教程如下：1、首先...

来自： pdcfighting的博客

289226
- DirectX修复工具增强版

最后更新：2018-12-20 DirectX修复工具最新版：DirectX Repair V3.8 增强版 NEW!

来自： VBcom的专栏

1697802
- 军事理论课答案（中国国防史）

中国国防史——秦至两晋南北朝已完成 成绩： 100.0分 1 【单选题】中国哪个历史...

来自： ling\_wang的博客

271870
- 100个小学生猜字谜大全及答案

100个小学生猜字谜大全及答案 1.字谜： 山上还有山。猜一字，答案是:出 2.字谜： ...

来自： 欢迎光临 包国...

208929
- tcp的java代码

服务器端 package com.car.client; import java.io.IOException; import java.io.Output...

来自： weixin\_4369414...

15081
- 学习Webpack（一）之 初识webpack

学习Webpack（一）之 初识webpack webpack简介 在官网中说，webpack是一个现...

来自： 幽幽小春

52



皮的开心

关注

原创16

粉丝4

喜欢0

评论2

等级： 博客之星

访问： 956

积分： 168

排名： 111万+

勋章： 1024

最新文章

自然语言处理学习（一）之概述小结

机器学习之路（八）随机森林

机器学习之路(七)支持向量机

机器学习之路(五)协同过滤及SVD算法

机器学习之路(四)之感知机算法PLA

个人分类

学习手册1篇

课程作业1篇

机器学习8篇

matplotlib2篇

linux2篇

展开

归档

2018年12月3篇

2018年10月9篇

2018年9月5篇

热门文章

关于linux模拟点击触摸屏操作的学习  
阅读量： 134

java实现网格法、KDTree空间检索  
阅读量： 107



阅读量：106

机器学习之路(七)支持向量机

阅读量：90

机器学习之路(二)之朴素贝叶斯

阅读量：79

最新评论

java实现网格法、KDTree空...  
asd2479745295: [reply]Wx975625[/reply] 没事啊  
可以直接用，我这里用的是jdk1.7

java实现网格法、KDTree空...  
Wx975625: 您好，我想问您个问题。import  
net.sf.json.JSONArray; 这个东西据说...

联系我们



微信客服



QQ客服

 QQ客服

 kefu@csdn.net

 客服论坛

 400-660-0108

工作时间 8:00-22:00

关于我们 招聘 广告服务 网站地图

 百度提供站内搜索 京ICP证09002463号

©1999-2018 江苏乐知网络技术有限公司

江苏知之为计算机有限公司 北京创新乐知  
信息技术有限公司版权所有

网络110报警服务 经营性网站备案信息

北京互联网违法和不良信息举报中心

中国互联网举报中心

0