**BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding**

**摘要** 我们介绍一种新的语言模型---bert，全称是双向编码表示Transformer。不同于最近的其他语言模型，bert基于所有层中的上下文语境来预训练深层的双向表示。因此，经过预先训练的bert表示仅需在最后一层输出层上进行微调，就可以胜任各项nlp任务，比如问题回答和语言推理，并不需要对各个不同的nlp任务进行大量模型上的修改。

Bert在概念上是很简单的，但是功能却很强大。它刷新了11个nlp任务的当前最好成绩，将GLUE的基准线提高到80.4%（提高了7.6%），将MultiNLI准确率提高到86.7%（提高了5.6%），将SQuAD v1.1问答测试的f1值提高到了93.2（提高了1.5%),比人类的水平高了2%。

**1. 引入**

近些年发现，预训练的语言模型在许多nlp任务中很有效。这些任务包括句子级任务，比如自然语言推理，句子释义（目的是通过对句子的整体分析来预测句子之间的关系），也包括词块级任务，比如命名实体识别，SQuAD问答任务（需要模型产在词块级别上生成细粒度输出）。

现有两种策略用于将预训练的语言模型应用于下游任务：基于特征以及微调。基于特征的方法，比如ELMo，使用了针对特定任务的模型架构（包括预训练表示来作为模型额外的特征）。至于微调方法，比如OpenAI GPT，引入了少量的特定任务的参数，并通过简单地微调预训练的参数来对下游任务进行训练。在先前的工作中，这两个方法在预训练阶段有相同的目标函数，即使用单向的语言模型来学习通用的语言表示。

我们认为，当前技术严重限制了预训练表示的能力，尤其是微调方法。其中最主要的局限是标准语言模型是单向的，这限制了在预训练中可以使用的模型架构类型。比如，在OpenAI GPT中，作者使用了一个从左到右的架构，每一个词块只能注意到transformer中self-attention层中的前面的词块。这些限制对于句子级别的任务而言是一个次优选择，但是对于词块级任务则是毁灭性打击。在词块级任务比如SQuAD问答任务中，结合两个方向上的语境是至关重要的。

在这篇论文中，我们通过bert（来自Transformer的双向编码器表示）改进了基于微调的方法。Bert

通过一个新的预训练目标（masked语言模型，由Cloze任务启发）解决了前面提到的单向限制。这个masked语言模型随机掩蔽一些输入中的词块，目标是基于这些遮蔽掉部分词块的输入来预测原始词汇id。不像从左到右语言模型的预训练，这个masked语言模型目标允许表示融合左右两侧的语境，这让我们可以预训练一个深层双向Transformer。除了masked语言模型，我们还引入了一个预测下一个句子的任务，这个任务联合训练文本对表征量。

我们论文的主要贡献如下：

1. 我们论证了双向预训练对语言模型的重要性，不同于Radford等人，他们使用了单向语言模型来进行预训练。Bert使用masked语言模型来实现预训练的深层双向表示。也不同于Peters等人，他们使用了从左到右和从右到左语言模型的浅层串联 。

2. 我们展现了预训练的表示能够消除许多大型工程制定特定任务架构的需求。Bert是第一个基于微调方法的表示模型并且它在多项句子级和词块级任务上刷新了记录，性能优于许多为某些特定任务而制定架构的系统。

3. Bert在11项nlp任务上刷新了记录。我们也对Bert进行部分切除，论证我们模型中的双向性质对整个模型的贡献最大。Bert的代码和预训练模型将会在goo.gl/language/bert上发布。

个人注解：谷歌提出的Bert与其他语言模型最大的不同点是Bert是一个真正意义上的双向模型，其他模型是单向模型（或者是两个单层的简单拼接）。

**2. 相关工作**

预训练的通用语言表示模型有很长的历史，本节我们主要回顾那些常用方法。

**2.1 基于特征的方法**

学习能够广泛应用的单词表示已经是一个数十年活跃的研究领域，其中既有非神经网络的方法，也有神经网络的方法。预训练的单词嵌入是现代nlp系统的必不可少的组成部分，这个方法与从头学习的单词嵌入相比性能上有着显著的提升。

这些方法已经被推广到更粗的粒度，比如句子嵌入或者是段落嵌入。跟传统的词嵌入一样，这些学习到的特征表示通常被用作下游模型的特征。

Elmo将传统的词嵌入推广到不同的维度。他们从语言模型中抽取语境敏感型特征。在将语境嵌入和现有的任务架构融合后，elmo刷新了一些nlp任务的基准线，比如SQuAD问题，情感分析，以及命名实体。

**2.2 微调方法**

最近出现了一个源于语言模型的迁移学习的新趋势，首先在一个LM目标上预训练一些模型架构，之后在针对某个有监督的下游任务进行微调。这些方法的优势是需要从头开始学习的参数几乎很少。由于这个，OpenAI GPT刷新了许多句子级别任务的GLUE基准线。

**2.3 在有监督的数据上进行迁移学习**

虽然无监督学习有一些优势，比如它可获得并使用的数据量几乎是没有限制的。但是也有一些研究表明从具有大量数据集的监督任务中也可进行有效的迁移学习。例如自然语言推理，机器翻译。在nlp之外的领域，比如计算机视觉，相关研究也表明了在大型预训练模型上进行迁移模型的重要性，其中一个很有效的方法就是微调在ImageNet预训练的模型。

个人注解：预训练语言模型在nlp任务中发挥着重要的作用，先前的word2vec，glove效果就很明显，但它们只是很浅层的词嵌入，预训练所得到的语言模型无法很好的处理比如多义词的问题，之后就出现比较深层的预训练语言模型，比如elmo，openai-gpt，bert。

**3. Bert**

在这一节中，我们将介绍Bert的实现。我们首先介绍bert的模型架构以及输入特征表示。之后，我们将在3.3节中介绍预训练任务，这也是本文的核心创新点。预训练程序和微调程序将在3.4节和3.5节中详细阐述。最后，在3.6节讨论Bert和OpenAI-GPT的区别。

3.1 模型架构

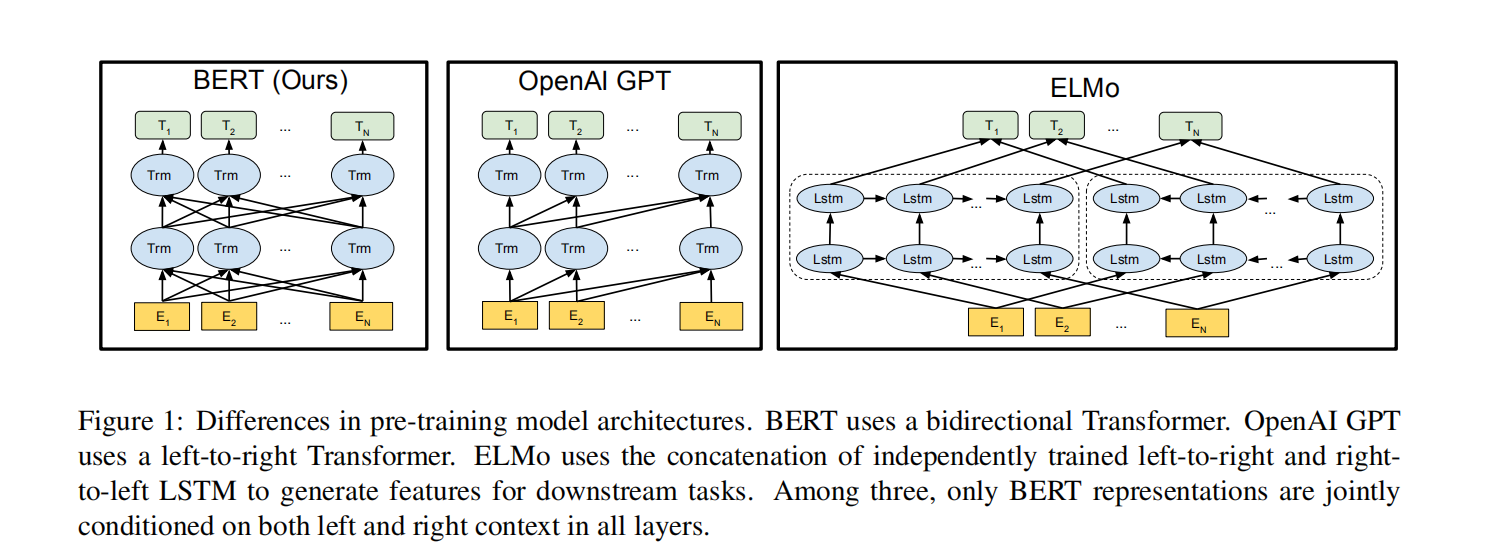
Bert模型架构是一个多层的双向编码Transformer（基于Vaswani等人描述并在tensor2tensor库中实现），由于Transformer最近相当流行，并且我们的实现和原始的Transformer很接近，因此我们会忽略模型模型架构详尽的背景描述，并且我们向读者推荐Vaswani等人的优秀指南，比如注释Transformer。

在Bert中，我们把层数(即Transformer块数)记为L，隐藏层大小记为H，self-attention头数量记为A。在所有情况下，我们设置前馈/过滤器的尺寸为4H，如H=768时为3072，H=1024时为4096。我们主要报告在两种尺寸模型上的结果。

（1）BertBase：L=12,H=768,A=12,总参数=110M

（2）BertLarge：L=24,H=1024,A=16,总参数=340M

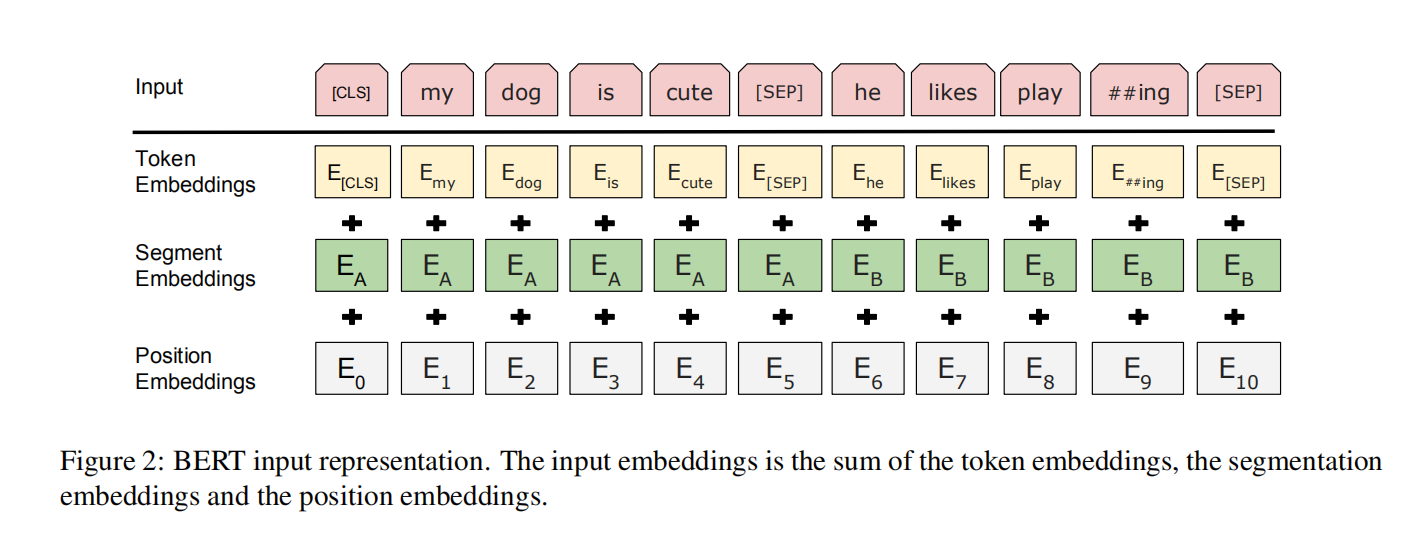
BertBase模型尺寸跟OpenAI GPT模型尺寸一样大，这样方便两个模型的比较。Bert跟OpenAI-GPT最重要的区别在于，bert的Transformer使用的时双向self-attention，OpenAI-GPT使用的受限制的self-attention，它的每个词块只能注意到其左侧的语境。我们注意到在文献中，双向Transformer常被指称为”Transformer编码器“，而其左侧语境版本常被称为”Transformer解码器“，因为它可用于文本生成。Bert，OpenAI GPT和ELMO之间的比较如图1所示。



个人注解：Bertbase于OpenAI-GPT的模型架构几乎一样，从图上可以看出，但Bert是双向的，而OpenAI GPT是单向的。

3.2 输入表示

我们的输入表示能够在一个词块序列中明确地表示单个文本句子或一对文本句子（比如句子问答任务）。对于一个给定的词块，它的输入表示可以由词块嵌入，段嵌入和位置嵌入求和而得到（见图2）



具体是：

1.我们使用了有30000个词块表的WordPiece嵌入，我们用##表示分词。

2.我们使用了位置嵌入，它所支持的最长长度是512。

3.每一个序列的第一个词块始终是特殊分类嵌入（CLS)，对应该词块的最后隐藏状态（Transformer的输出）被用作分类任务的聚合序列表示，对于非分类任务，这个向量会被忽略。

4.句子对被打包成单个序列，我们以两种方式来区分句子。首先，我们用一个特殊的词块[SEP]将他们分开。其次，我们添加了在第一个句子的每个词块中添加句子A嵌入，在第二个句子的每个词块中添加句子B嵌入。

5.对于单据输入，我们只是用句子A嵌入。

**3.3 预训练任务**

与Peters等人和Radford等人的方法不同，我们不适用传统的从左到右或从右到左的语言模型来预训练Bert。相反，我们使用两个新的无监督预测任务来语寻来Bert，如本节所述。

**3.3.1 任务一：Masked 语言模型**

直观的说，我们有充分的理由相信深层双向模型的性能要远远比由左向右模型或者是简单的将由左向右和由右向左模型浅层拼接起来而得到的模型强。不幸的是，标准的条件语言模型只能从左到右或从右到左进行训练，因为双向条件会允许每个单词在多层语境中间接地看到自己。

为了训练一个深层的双向表示，我们选择了一种直接的方法，随机遮蔽一些输入的词块，之后仅仅预测这些遮蔽掉的词块。我们将这个过程称为”masked LM”(MLM), 尽管它在文献中通常被称为完形填空任务。在这种情况下，对应于遮蔽词块的最终隐藏向量送入词汇表上的softmax中，正如在标准语言模型所做的一样。在我们所有的实验中，我们随机遮蔽掉每个序列中所有WordPiece词块的15%。与去噪声的自编码器相反，我们只预测被遮蔽掉的词块，而不是重新预测整个输入。

尽管这个确实让我们获得一个预训练的双向模型，这么做仍然由两个缺点。第一个缺点是我们制作了预训练和微调的不匹配，因为[MASK]词块在微调时从未被看见。为了缓解这个问题，我们并不总是用[MASK]词块替换掉遮蔽单词。相反，训练数据生成器随机选择15%的词块。比如，在句子我的狗是毛茸茸的，生成器选择了毛茸茸的，之后完成以下过程。

\* 生成器并不总是用[MASK]去替换选择的词，它做一下操作：

\* 80%的时间：用[MASK]替换选择的词，比如我的狗是毛茸茸的！我的狗是[MASK]。

\* 10%的时间：用随机词体替换选择的词，比如我的狗是毛茸茸的! 我的狗是苹果。

\* 10%的时间：让选择的词保持不变。比如我的狗是毛茸茸的！我的狗是毛茸茸的。这么做的目的是将表示偏向于实际观察到的单词。

Transformer编码器不知道下一个它要预测的单词已经被随机单词替换，因此它会被迫保持每个输入词块的分布式的语境表示。另外，由于随机替换在所有词块中只有1.5%的几率发生，着不会对模型的语言理解能力产生影响。

第二个缺点是使用MLM之后，每个批次只有15%的词块被预测，为了让模型收敛，需要更多的预训练步骤。在5.3节中，我们证明了MLM的收敛速度略慢于从左到右的模型（从左到右模型预测每一个词块），但是MLM模型性能上的提升远远超过所增加的训练成本。

个人注解：由于要训练一个深层的双向语言模型，我们不能沿用传统语言模型的训练策略，作者采用了一个很巧妙的方法，随机MASK某些单词来训练双向模型，跟完形填空任务如出一辙。与此同时，带来了两个弊端，一是MASK策略会使得预训练和微调不匹配，二是训练时间的增加。

**3.3.2 任务2：下一句预测**

很多重要的下游任务比如为问答(QA)和自然语言推理（NLI)，都是基于对两个句子间关系的理解。为了训练一个能够句子关系的模型，我们预训练了一个二值化下一个句子的预测任务，这个任务可以很简单的从单词语料库中生成。具体来说，我们首先选择句子A和B作为预训练样本：B由百分之五十的可能性是A实际的下一句，也有百分之五十的可能性是语料库中的随机句子。比如：

输入=[CLS] the man went to [MASK] store [ SEP] he bought a gallon [MASK] milk [SEP]

标记=IsNext

输入=[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flight #less birds [SEP]

标记=NotNext

我们完全随机的选择NotNext标记的句子，最后预训练的模型能在这个任务上达到97-98%的准确率，尽管这个任务很简单，我们在5.1节将证明，这个任务的预训练对QA和NLI都非常有用。

**3.4 预训练过程**

Bert预训练过程主要遵循了文献上的语言模型预训练过程。对于预训练语料库，我们使用了BooksCorpus（800M单词）和英文维基百科（2500M单词)的串联串接。对于维基百科，我们只抽取文本段落，忽略了列表，表格和标题。很关键的一个要点是，使用文档级别的语料库而不是洗牌式句子级别的语料库，比如Billion Word Benchmark，这样便于抽取长的连续序列。

为了生成每个训练输入序列，我们从语料库中采样了两个文本跨度，我们将其称为句子，尽管他们通常比单个句子长的多（也可能短得多）。第一个句子采用A嵌入，第二个句子采用B嵌入。B有50%的可能性是A下一个句子，有50%的可能性是一个随机句子，这专门是为了”预测下一个句子“任务而设置的。对它们进行采样，使得它们的组合长度<=512个词块，LM遮蔽应用于WordPiece词块化之后，有15%的词块被遮蔽，并且不特别考虑部分词块。

我们训练批量的大小为256个序列（256个序列\*512个词块=128000个词块/批次），训练了总计1000000步，大约是在33忆个单词语料库上训练了40个周期。我们使用了Adam，学习率设置为1e-4，β1为0.9，β2为0.999，L2权重衰减为0.01，学习率首先预热10000步，之后线性衰减学习率。我们在所有层上使用0.1的丢失概率。在OpenAI GPT之后，我们使用了gelu激活而不是标准的relu激活。训练的损失函数是遮蔽LM的平均可能性和下一个句子预测可能性的总和。

Bertbase的预训练是在Pod配置的4个云TPU上进行（总计16个TPU芯片）。BertLarge的训练是在16个云TPU上进行（总计64个TPU芯片），每次预训练需大约4天完成。

**3.5 微调过程**

对于序列级分类任务，BERT的微调很简单。为了获得输入序列的固定维度的池化特征表示，我们取第一个词块输入的最终隐藏状态（Transformer的输出),它通过特殊词嵌入[CLS]来构造。将该向量表示为C∈,

。微调期间添加的唯一新参数是分类层向量,这里K为分类标记的数量。标记概率是通过softmax计算，即。BERT所有的参数和W都经过联动地微调，以最大化正确标记地对数概率。对于跨度和词块级地预测任务，必须以任务特定方式稍微修改上述过程，具体细节在第四节地各个小节中给出。

对于微调，大多数地模型超参数和预训练地相同，除了批量大小，学习率，训练周期数。丢失概率总是保持在0.1。最佳地的超参数值是跟特定任务相关的，但是我们发现以下的值可能在所有任务能够很好的工作。

批量大小：16，32

学习率：5e-5，3e-5，2e-5

周期数：3，4

我们也发现对于大数据集(100k+词块的训练样例)，超参数选择的敏感型远小于小数据集。微调通常非常快，因此有理由对上述超参数进行详尽的搜索，并且选择在开发集上性能最佳的模型。

**3.6 BERT和OpenAI GPT比较**

在当今所有存在的预训练模型中，OpenAI GPT（它在大型文本语料库上从左到右训练Transformer LM）与BERT最具有可比性。实际上，许多BERT的设计决策被有意地选择以尽可能地接近GPT,这样做能够最细微地比较这两种方法。这项工作的核心是要论述3.3节中两个新型预训练任务对模型性能提高起了主要作用。但同时我们注意到BERT和GPT在训练时还有一些其他的差异：

\* GPT的语料库是BooksCorpus（800M单词），BERT的语料库是BooksCorpus（2500M单词)

\* GPT使用的句子分隔符（[SEP])和分类符词块([CLS])仅在微调时引入。BERT在预训练期间学习[SEP],[CLS]和句子A/B嵌入。

\* GPT用一个批量32000单词训练1M步；BERT用一个批量128000单词训练1M步。

\* GPT对所有微调实验使用了5e-5相同学习率，BERT选择特定于任务的微调学习率（在开发集上最佳的学习率）

为了分离这些差异的影响，我们在5.1节进行了消模实验，证明性能的提升主要来自于新型预训练任务。

个人注解：作者设计的对比实验很严谨，列出了所有不同的参数变量。但同时我们可以很明显，BERT的模型规模要远大于OPENAI-GPT,这也是深度学习的趋势。

**4 实验**

在这节中，我们将介绍BERT在11个任务上的微调结果。

**4.1 GLUE数据集**

通用语言理解评估（GLUE)基准是多种自然语言理解任务的集合。大多数GLUE数据集已经存在多年，GLUE的目标是将Train，Dev和Test规范的拆分并发行这些数据集，除此之外，设立评估服务器来降低评估不一致以及测试集过拟合的可能性。GLUE并不会为测试集分发标签，用户必须将他们的预测上传到GLUE服务器来进行评估，并且上传的次数有限制。

GLUE基准包括下列数据集，它们的描述最初在Wang等人的文章中进行了总结。

\* MNLI 多类型自然语言推理是一个大规模的众包的蕴含分类任务。给出一对句子，目标是预测第二个句子和第一个句子相比是蕴含，矛盾，还是中立。

\* QQP Quora问题对是一个二值的单句子分类问题，目标是判断两个在Quora上的问题是否在语义上等价。

\* QNLI 问题自然语言推理是斯坦福问答数据集的一个版本，它被转换成二值分类问题。正例是指那些问题，句子对包含正确的答案，反例是指那些问题，句子对来自于同一段落，但不包含正确的答案。

\* SST-2 斯坦福情感树库是一个二值的单句子分类问题，它包括许多影评中的句子，这些句子中包含着人们对于他们情感的注释。

\* COLA 语言可接受性语料库是一个二值的单句子分类问题，它的目标是预测一个英文句子是否语法上是否有问题。

\* STS-B 语义文本相似性基准是一个句子对的集合，它从新闻提要和其他地方提取，他们被标记为1到5分，来表示两个句子语义上是否相近。

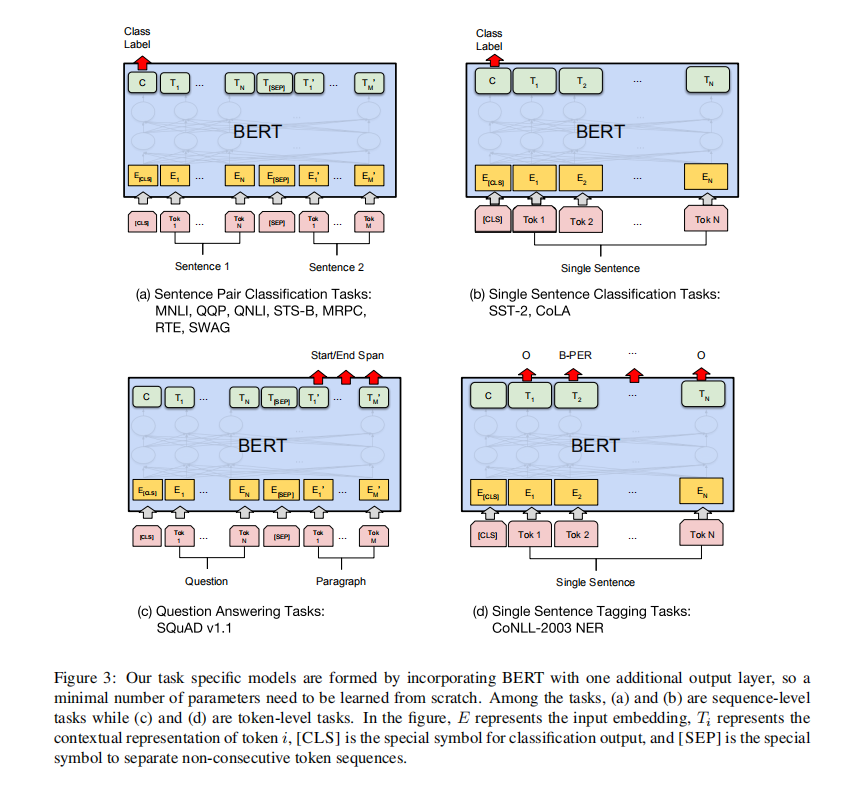
\* MRPC 微软研究院解释语料库包含了从网上新闻中自动抽取的句子对，人工标注句子对是否语义上相等。

\* WNLI 维诺格拉德自然语言推理是一个源自（Levesque等，2011）的小型的自然语言推理数据集。Glue网站上指出这个数据集的构建有问题，并且每个提交给GLUE的训练系统总是比预测大多数类别的65.1的基准线准确度要差。因此，出于对OpenAI GPT的公平性，我们将这一项任务排除在外。在GLUE上提交时，我们总是预测大多数类别。

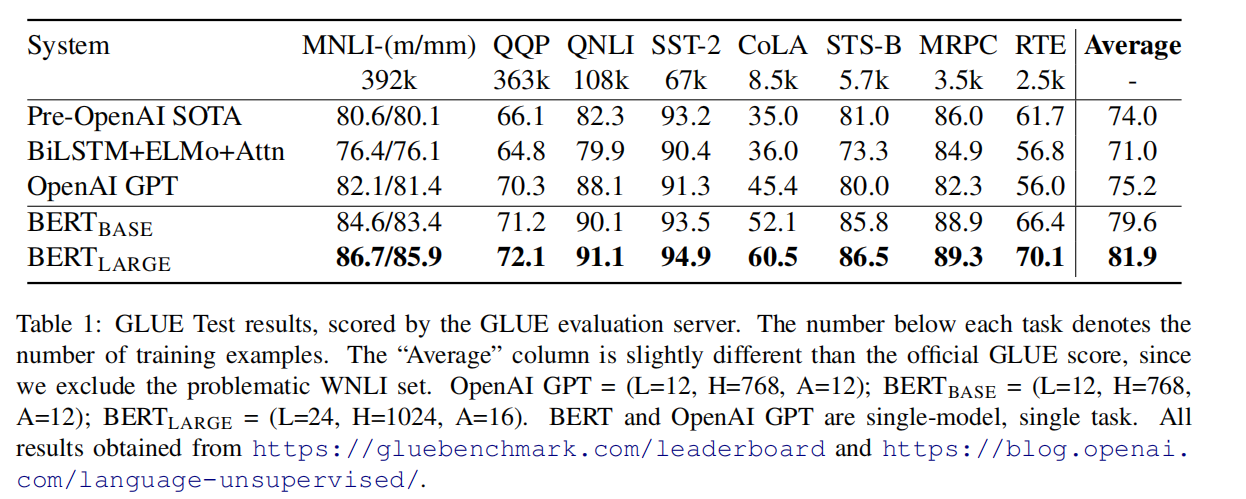
**4.1.1 GLUE结果**

为了在GLUE上进行微调，我们将输入序列或者序列对表示成第3节中所描述的样子，并且使用了最后隐藏向量来表示第一个输入词块（[CLS])，以此作为聚合特征表示。这些都呈现在可视化图3(a)和(b)中。唯一在微调中引入的新参数是分类层，这里K是标签的数量。我们用C和W计算标准分类

损失，比如。



在所有GLUE任务上，我们的批量大小为32，周期为3。对于每项任务，我们将微调的学习率分别设置为5e-5，4e-5，3e-5和2e-5，选择在开发集上表现最好的超参数。另外，对于BertLarge，我们发现有时候微调在小数据集上不太稳定，因此我们重新随机运行了多次模型，选择在开发集上最佳的模型。在每次随机运行时，我们选择相同的预训练检查点，但是微调数据混洗和分类层初始化不同。GLUE数据集不包括其测试标签，BertBase和BertLarge只在GLUE服务器上提交了一次。



实验结果在表格1中。BertBase和BertLarge都在所有任务上远远超过了之前的系统，相较于最先进水平，平均准确率提高了4.4%和6.7%。注意到BertBase和OpenAI GPT除了注意力遮蔽之外在模型架构上几乎相同。对于规模最大，报道最广泛的GLUE任务，MNLI，BERT领先最先进水平4.7个百分点的准确率。在官方GLUE排行榜上，GlueLarge得分为80.4，而曾经在该排行系统上登顶的系统OpenAI GPT在本文撰写之日的GLUE得分为72.8。

有趣的时，BertLarge在所有任务上都明显由于BertBase，即使对于训练集非常少的任务。我们将在5.2节更全面的讨论Bert模型尺寸的效应。

**4.2 斯坦福问答数据集v1.1（SQuAD)**

SQuAD使一个包含100k众包问答对的集合。给出一个问题和维基中包含回答的一个段落，这个任务的目标时预测出段落中的回答文本跨度。比如：

\* 输入问题

Where do water droplets collide with ice crystals to form precipitation？

\* 输入段落

...Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud....

\* 输出回答

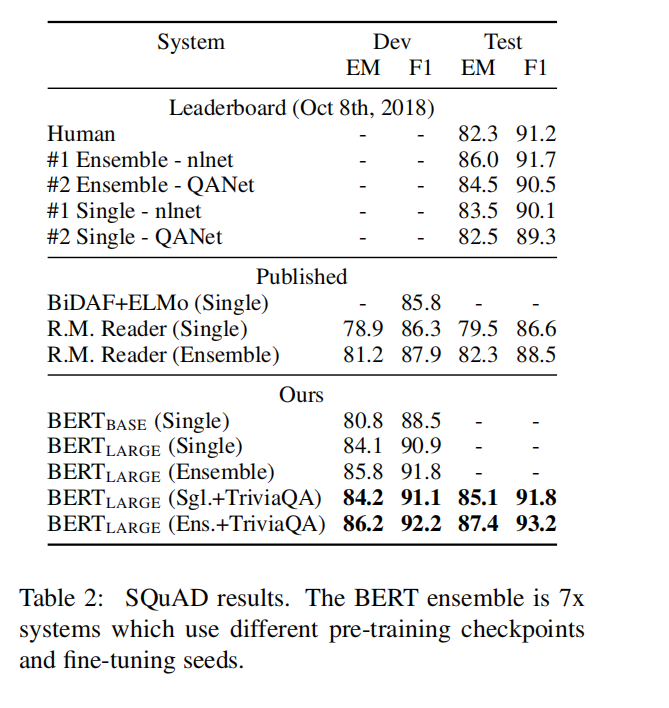
Within a cloud

这种类型的跨度预测任务和GLUE中的序列分类任务完全不同，但是我们能够调整Bert，使得它在SQuAD上运行。跟GLUE一样，我们将输入的问题和段落表示成单个打包序列，问题使用A嵌入和段落使用B嵌入。微调期间唯一的新参数时开的向量，结束向量。让BERT对应第i个词块输入的最后的隐藏向量表示成,具体的可以看可视化图3（c)。之后，单词i作为答案跨度开始的概率被计算为Ti和S之间的点积，之后跟着一个段落中所有单词的softmax:



相同的公式被用于答案跨度的末端，最大评分范围被用于预测。训练目标时正确开始和结束位置的log似然函数。

我们将学习率设置为5e-5，批量大小设置为32，训练了3个周期。再推断的时候，因为对于文本跨度结束位置预测不依赖于开始位置，我们增加一个限制，即结束位置必须在开始位置之后，除此之外，没有任何启发式信息被使用。将标记的词块化跨度和原始未标记的输入对齐，以便于评估。



实验结果呈现在表2之中，SQuAD使用了一个非常严格的测试流程，提交者必须自己联系SQuAD组织，在不公开的测试集上跑模型并得到结果，因此我们只能够提交我们最好的系统用于测试。表格中显示的结果是我们第一次且唯一一次SQuAD提交的结果。我们注意到SQuAD排行榜上最好的结果没有最新的可用公共系统描述，并且在训练系统时可以使用任何公共数据集。因此我们在我们提交的系统上使用非常适度的数据增强， 在SQuAD和TriviaQA上联合训练。

我们性能最好的系统优于排行榜上第一的系统，融合系统提升了1.5个F1项，单一系统提升了1.3个F1项。事实上，我们的单模型BERT在F1值上优于排名第一的系统(它是经过融合而得的系统)。如果我们仅仅在SQuAD上微调，我们在f1值上会降低0.1到0.4，但是仍然远远超越已存在的任何系统。

**4.3 命名实体识别**

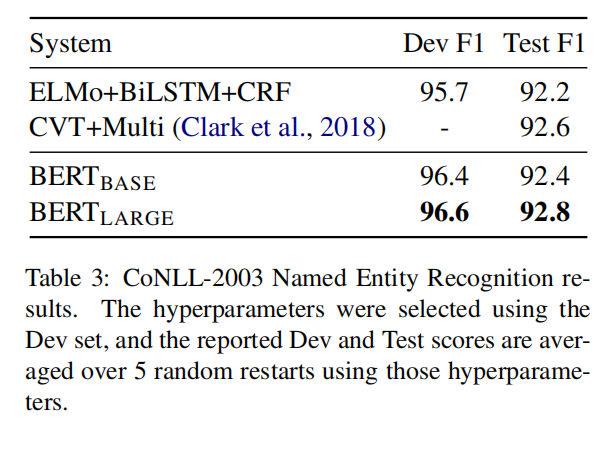
为了评估词块标记任务的功能，我们在CoNLL 2013 命名实体识别数据集上进行微调。这个数据集包括了200K训练单词，它们被标记成人员，组织，地点，杂项或其他（非命名实体）。

为了进行微调，我们将最终隐藏特征表示提供给每个词块i到NER标签集上的分类层。每个单词的预测并不依赖于周围的预测（即不是自回归也没有CRF)。为了使这个于WordPiece词块兼容，我们将每个CoNLL词块化单词输入到我们的WordPiece词块化器，并且使用与第一个子标记相关的隐藏状态作为分类器的输入。比如：

Jim Hen ##son was a puppet ##eer

I-PER I-PER X O O O X

对X没有做预测，因为WordPiece词块边界是输入的已知部分，对训练和预测都做了类似的处理。可视化图在图3（d）中呈现。一种事例WordPiece模型被用于NER,而非事例模型被用于其他任务之中。



实验结果在表格3呈现，BERTLarge性能超越了现有的SOTA--一种具有多任务的跨试图训练，在CoNLL-2003 NER测试中领先了大约0.2个f1值。

**4.4 对抗生成情景数据集**

SWAG数据集包含113k个句子对的完整样例，用于评估基础常识推理。

给定视频字幕中数据集的某个句子，任务的目标是在四个选项中选择最合理的后续。比如：

A girl is going across a set of monkey bars. She

(i) jumps up across hte monkey bars

(ii) struggles onto the bars of grab her head.

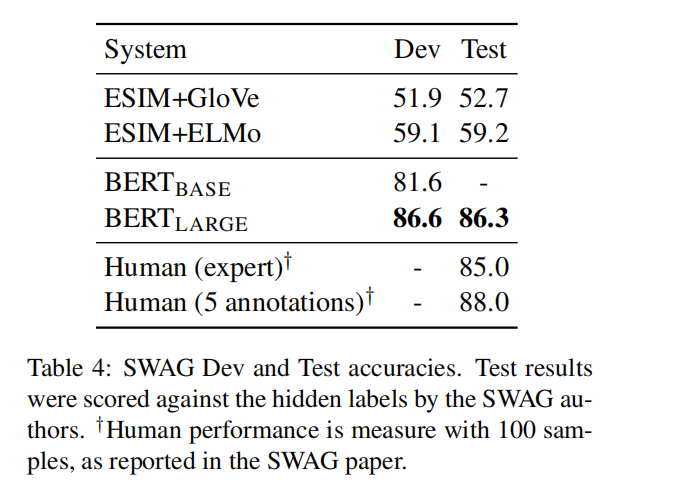
(iii)gets to the end and stands on a wooden plank

(iv)jumps up and does a back flip.

将BERT应用到SWAG数据集的过程跟GLUE很类似。对于每一个样本，我们构造四个输入序列，每一个序列是句子A和可能的后续句子B的串联。我们引入的唯一任务特定参数是一个向量,它和最后聚合特征表示的点积代表每个选择i的得分，概率分布是四个选择的softmax：



我们设置学习率为2e-5，批量大小为16，微调3个周期。结果被呈现在表4中。BertLarge超出了作者给出的基准线（ESIM+ELMo系统)大约27.1个f1值.



**5 消模实验**

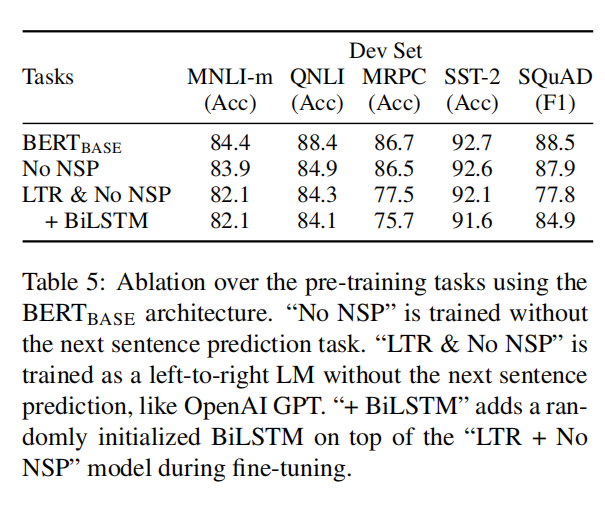
虽然我们已经给出了Bert十分出色的实验结果，但到目前为止，我们并没有分离BERT框架的各个组件，以比较它们各自对模型的贡献。在这一节中，我们将对BERT的许多组件进行分离以更好地理解他们的相对重要性。

**5.1 预训练任务的影响**

我们的核心主张之一是BERT的深层双向模型架构，通过masked语言模型预训练而得，它是BERT与以前工作相比最为重要的改进点。为了支撑这个论点，我们评估了两个模型，它们使用相同的预训练数据集，微调策略以及Transformer参数（跟BertBase相同）：

1. 无NSP:一个使用了MLM任务但是没有NSP任务的模型

2.LTR & No NSP:一个使用了LRT而不是MLM的模型，在这种情况下，我们预测每个输入单词，不遮蔽任何单词。左侧约束也适用于微调（因为我们发现在预训练上只适用左侧语境，微调时候使用两侧语境会使得模型的效果很差）。另外，这个模型是在NSP任务的情况下预训练。这与OpenAI GPT直接相当，但使用了更大的训练集，我们自己的输入特征表示以及我们自己的微调方案。



实验结果记录在表格5中，我们首先观察NSP任务带来的影响，我们可以看见移除NSP严重影响了在QNLI,MNLI和SQuAD任务上的性能。这些结果表明我们的预训练方法对于互动二先前提出的强有力的实证结果至关重要。

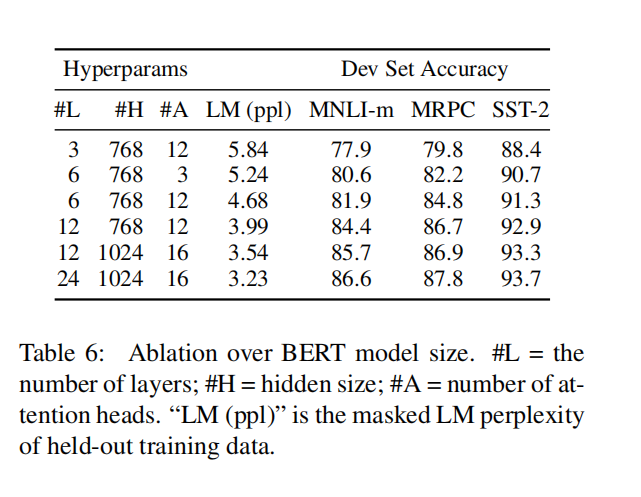
接下来，我们通过比较“No NSP”和”NTR & No NSP”来评估双向表示的影响。LTR模型在所有任务上的性能都比MLM模型要差，尤其是在MRPC和SQuAD任务上。对于SQuAD可以很明显的发现LTR模型在跨度和词块预测方面性能非常差，因为其词块级别隐藏状态没有右侧语境。对于MPRC，目前尚不清楚性能不佳是否是由于其数据量较小还是该任务的本质，但是我们发现即使通过很多次的随即重启来进行完整的超参数扫描，模型的性能还是比较差。

为了尝试增强LTR系统，我们在上面加了一层随机初始化的BiLSTM用于微调。这确实大大增强了在SQuAD上的表现，但是与预先训练的双向模型相比，性能还是差了一截。它还影响了所有四个GLUE任务的性能。

我们认识到，也可以独立训练LTR和RTL模型，并将每个词快表示为这两个模型的串联，跟ELMo一样。但是：（a）所花费的代价是单向模型的两倍（b）对于像QA这样的任务来说，这不是很直观，因为RTL模型无法对问题的答案作出规定(c)它的强度远远低于深层双向，因为深层双向模型可以选择使用左侧语境还是右侧语境。

**5.2 模型大小的影响**

在这小节中，我们将探索模型大小对于微调任务准确率的影响，我们训练了许多BERT模型，它们的层数，隐藏层的大小，注意力头的数量各不相同。与此同时，我们使用跟前文描述的相同的超参数和训练过程。

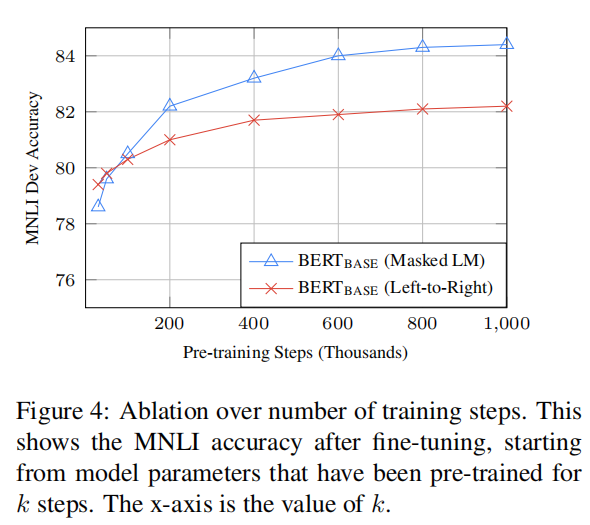


在部分GLUE任务上的实验结果在表格6中呈现。在这个表格中，我们展现了随机五次重启后在开发集上取平均准确率的微调结果。我们可以发现模型越大，在所有四个数据集上的准确率就越高，即使是对于MRPC任务(这个任务只有3600个标记的训练集，同时它跟其他预训练任务有很大的不同）。同时令人惊讶的是，我们能够在相对于现有文献已经相当大的模型上取得如此惊人的提升。例如，Vaswani等人探索的最大的Transformer是有100M个参数(L=6, H=1024,A=16)的编码器。迄今为止，我们在文献中能够查到的最大的Transformer是一个有235M个参数(L=64, H=512, A=2)的编码器。与之相对应，BertBase包含了110M个参数，BertLarge包含了340M个参数。

众所周知，增大模型的尺寸将会在机器翻译和语言模型等大型任务上取得性能上的持续提升，这个可以通过表6观察到。但是，我们相信这是第一次被证明将模型扩展到超大的尺寸也会导致在非常小规模的任务上实现性能上的大幅提升，当然前提是该模型已经得到充分的预训练。

**5.3 训练步数的影响**

图4呈现了从已训练k步的检查点微调之后的MNLI Dev准确率，这使得我们能够回答下列问题：



1.问题：BERT是否真的需要长时间的预训练（128000字/批\*1000000步）来实现很高的微调准确率

回答：是的，拿1M步和500K步作比较，BertBase在MNLI上获得了接近1%的额外准确率。

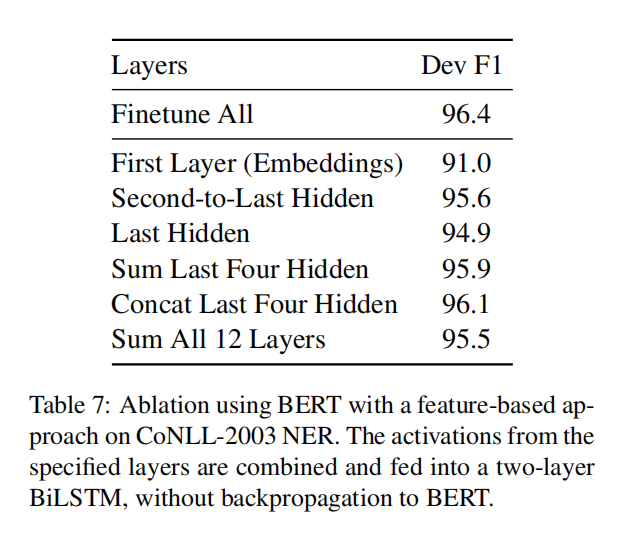
2. 问题：MLM预训练是否收敛的比LTR预训练慢？由于LTR在每个批次只预测15%的单词，MLM预测每个单词。

回答：MLM模型收敛的的确比LTR模型慢一些。但是，在绝对准确率上，MLM模型几乎一开始就超越了LTR模型。

**5.4 基于特征的BERT方法**

至今为止，所有呈现的BERT结果都是通过微调方法，即在预训练模型后面加一层简单的分类层，在下游任务上所有的参数同时微调。但是，基于特征的方法，即从预训练模型中抽取固定特征，也有某些优点。首先，不是所有的NLP任务可以被Transformer编码器架构很好的表示，因此需要一个添加一个额外的特定任务的架构。其次，如果能够预计算一个训练集的特征表示，就能够在这个特征表示上以很低的代价跑多次实现，这会带来许多好处。

在这一节，我们通过在CONLL-2003 NER任务上产生跟ELMO类似的预训练的语境特征表示，来评估BERT在基于特征方法上的表现。为了实现这个，我们使用了4.3节中描述的相同的输入特征表示，但是使用了一层或多层激活函数之后的特征表示，并没有微调BERT上的任何参数。在分类层之前，这些语境嵌入层被用作两层768维BiLSTM随机初始化的输入。



结果在表格7中呈现出来，性能最佳的方法是连接来自预训练Transformer顶层四个隐藏层的词块特征表示，它只比微调整个模型差了0.3个F1值。这表明BERT对于微调和基于特征的方法都是有效的。

**6 结论**

近期关于迁移学习的语言模型的改进实验论证了大量的，无监督预训练是许多语言理解系统的集成部分。特别是，这些结果使得低资源任务也能从非常深的单向架构中受益。我们主要的贡献是进一步将这些发现推广的深层双向模型架构，使得相同的预训练模型能够成功的解决一系列广泛的NLP任务。

实验的结果非常出色，在某些情况下超越了人类的表现。未来的重要工作是研究BERT能否捕获语言现象。