# Title：

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

深的双向的预训练Transformer针对语言理解任务

在CV，很早的时候就可以在一个大的数据集上（ImageNet）训练一个CNN的模型，这个模型可以帮助很多视觉的任务，在BERT之前一直没有一个深的神经网络能够帮助NLP的任务，BERT的出现弥补了这一点。

# Abstract：

BERT的意思是Bidirectional Encoder Representations from Transformers：Transformer模型的双向编码表示，和ELMO，GPT不一样，BERT是用来训练深的双向的表示（用左边的和右边的信息预测中间的），使得它可以只添加一个额外的输出层就可以在很多NLP的任务上面达成很好的效果

# Introduction

在语言模型内，预训练可以提升很多任务的效果，包括两类：一是句子层面的任务，比如对句子情绪的识别或者两个句子之间的关系；第二是词元层面（token-level）的任务，比如实体命名的识别（实体命名：人名、街道名等）。

在使用预训练模型做特征表示的时候，有两类策略：第一是基于特征的（feature-based），第二是基于微调的（fine-tuning）。第一类的代表性工作是ELMo（根据下游任务构造一个和它相关的神经网络），预训练好的表示跟输入一起进入网络，作为一个很好的特征表达。第二类是GPT：预训练好的权重根据下游任务数据进行微调。

这些方法会有局限性，标准的语言模型是单向的，导致在选择架构的时候会有局限性：GPT从左到右。如果要做句子层面的分析，从左看到右和一次性看所有是合法的，就算是词元上的任务（Q&A），也是合法的，把两个方向的信息都放进来因该是能够提升任务的性能。

BERT使用了一个带掩码的语言模型（MLM）。每次随机选择一个词元把它盖住，目标函数是预测那些被盖住的字（完形填空），这样就能够训练一个深的双向语言模型。他还可以做“下一个句子的预测“：给两个句子判断这两个句子在原文中是不是相邻的，这样能够让模型学习到一些句子层面的信息。

贡献：

1．展示了双向信息的重要性

2．一个很好的预训练模型，这样就不用对特定任务改动模型了，BERT是第一个微调模型在很多NLP任务上都取得了很好的成绩

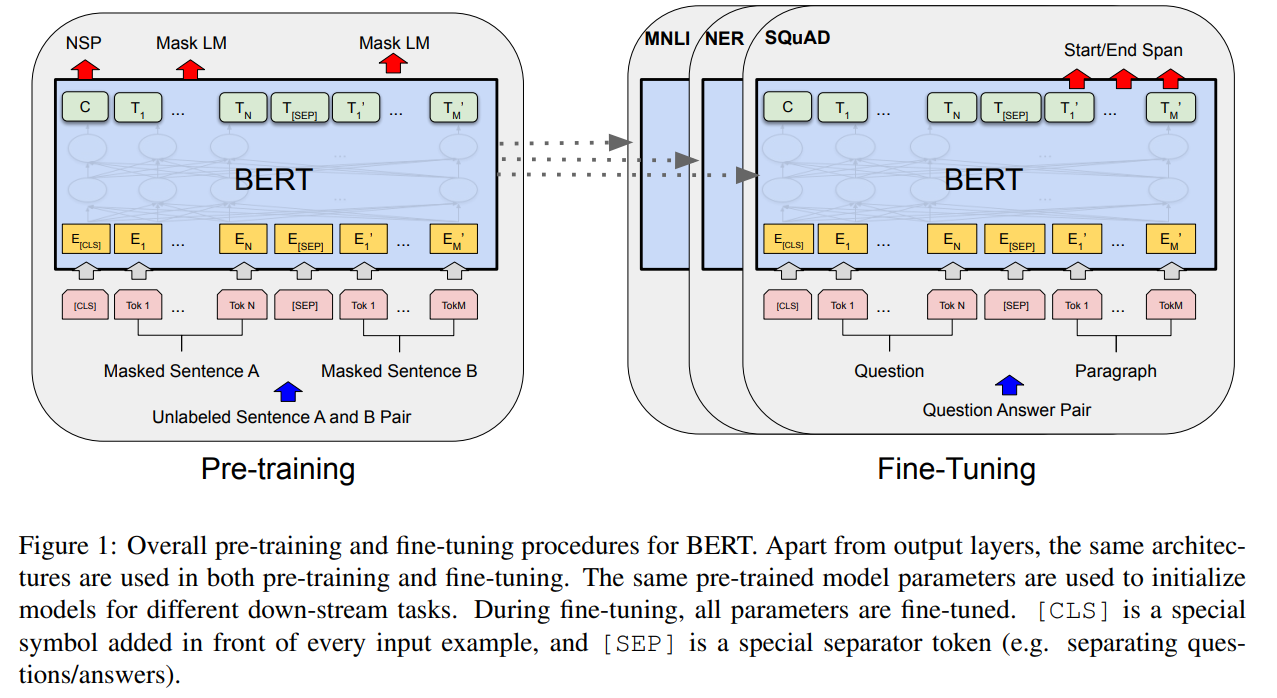
3．开源

# Conclusion：

使用非监督的预训练能够使得训练样本比较小的任务也能够享受到深度神经网络，作者的主要贡献是把前人的工作拓展到深的双向的架构上面，使得同样一个预训练模型能够处理大量的自然语言任务。

# BERT

BERT有两个步骤：预训练和微调。预训练是在没有标号的训练集上训练的，微调时是把预训练的权重作为初始化然后再在有标号的数据上进行训练，每一个下游任务都能创建一个新的BERT模型。



**Model Architecture**

模型结构是多层的双向的Transformer编码器，没有什么改动。

作者调整了三个参数，第一个是L（Transformer块的个数），第二个是H（隐藏层大小），第三个是A（自注意力机制的多头的个数），分别有两个模型：BERT-BASE (L=12, H=768, A=12, Total Parameters=110M) and BERT-LARGE (L=24, H=1024,A=16, Total Parameters=340M).

如何把超参数换算成可学习参数的大小？

Transformer模型的可学习的参数主要来自两块：嵌入层和Transformer块。

嵌入层是个矩阵，输入是字典的大小（30k），输出等于隐藏单元的个数为H。可学习参数为30k\*H。输出会进入Transformer块 。

Transformer块有两个模块，第一个是多头注意力，第二个是MLP。自注意力机制本身没有可学习参数，但是对于多头注意力，它会分别对Q、K、V做一次投影，每个投影的维度是64，总共有A个头，A\*64=H。每个Q、K、V都有A个投影矩阵，合并起来就是H\*H的矩阵，输出后还会做一次投影，加上输出的H\*H的矩阵就是4\*H^2个参数。然后进入MLP，MLP里有两个全连接层，第一个输入是H，输出为4\*H，第二层输入是4\*H输出是H，所以加起来是4\*H\*H+4\*H\*H=8\*H^2。

一个Transformer块可学习的参数为12\*H^2。共有L个，故可学习参数为12\*L\*H^2。

总共可学习参数就是：30k\*H+12\*L\*H^2

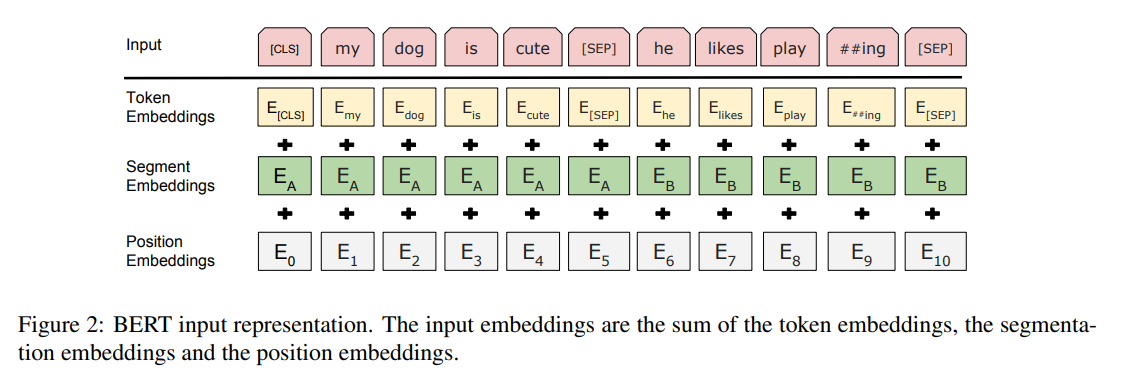
**Input/Output Representations**

不同的下游任务需要处理不同长度的句子。所以BERT的输入是一个序列（Transformer是一个序列对分别进入编码器和解码器）。切好序列，让30k的词典能够表示很大的文本.

如何把他们放在一起呢？第一个词是[cls]，特殊的记号代表classification token，BERT希望它的输出代表最后整个序列的输出。因为它要把两个句子放在一起，所以为了区分他们，BERT在句子末尾放上一个[seq]：代表separate。

或者学一个嵌入层，表示它是第一个句子还是第二个句子。

对于每个token，进入BERT的向量表示是词元的embedding+句子的embedding+位置embedding（可学习）

嵌入层做法：给定token的序列输出向量序列，进入Transformer块

## Pre-training BERT

**Task #1: Masked LM**

mask15%的token，在这15%的token内，80%被替换成[mask]，10%替换成词典内随机词，10%什么都不做

**Task #2: Next Sentence Prediction (NSP)**

在QA和NLI自然语言推理内都是一个句子对，如果能让它学习一些句子层面的信息是不错的。在句子对中有50%的概率A和B是顺序的两个句子，50%的概率是随机选的句子（一半样本是正例，一半是负例）

**Pre-training data**

BooksCorpus (800M words) and English Wikipedia (2,500M words).

## Fine-tuning BERT

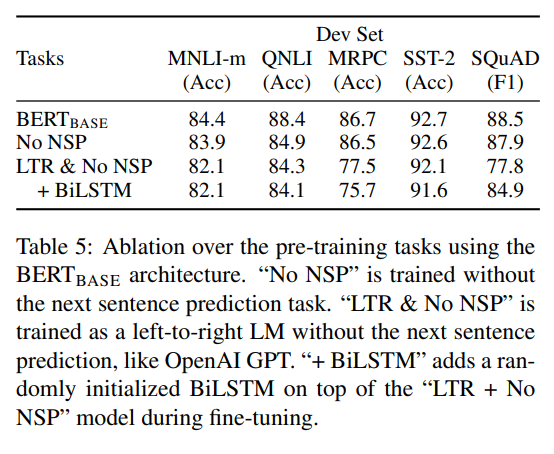
因为BERT的编码器输入是双向的，故可以看到双向的信息，但也损失了解码器，不能做机器翻译了。

在做下游任务的时候，根据任务做相关的输入和输出，根据下游任务的不同，要么是拿到cls做分类，要么加个softmax做输出

# Experiments

详细看论文

# Ablation Studies



如果是从左看到右会怎么样

# Question：

**1. BERT分为哪两种任务，各自的作用是什么；**

Masked Language Model：这种任务的目的是预测句子中部分单词的原始形式。在训练过程中，BERT模型会随机选择一些单词并用“【MASK】”标记替换它们。模型的任务是预测被替换的单词的原始形式。这种方法可以使模型在理解句子语义的同时学习到词语之间的关系。

Next Sentence Prediction：这种任务的目的是预测一个句子是否是另一个句子的下一句。在训练过程中，BERT模型会从两个句子中选择一个随机的句子对，并根据是否是下一句来训练模型。这种方法可以使模型更好地理解上下文之间的关系。

**2.在计算MLM预训练任务的损失函数的时候，参与计算的Tokens有哪些？是全部的15%的词汇还是15%词汇中真正被Mask的那些tokens？**

只有真正被替换成了“【MASK】”标记的tokens会被用来计算损失值，而没有被标记的tokens则不会参与损失函数的计算。

**3.在实现损失函数的时候，怎么确保没有被 Mask 的函数不参与到损失计算中去；**

在实现损失函数时，需要使用一个掩码（mask）向量来指示哪些tokens是被Mask的，哪些tokens是没有被Mask的。具体来说，掩码向量中被Mask的tokens的位置为1，没有被Mask的tokens的位置为0。在计算损失函数时，可以将掩码向量与预测的tokens和实际的tokens相乘，这样就可以将没有被Mask的tokens的损失值置为0，只计算被Mask的tokens的损失值。

以PyTorch为例，可以使用以下代码来实现掩码的功能：

loss\_mask = torch.tensor(mask, dtype=torch.float32) # mask是掩码向量

predictions = model(tokens) # tokens是输入的tokens

loss = loss\_function(predictions, labels)

masked\_loss = torch.sum(loss \* loss\_mask) / torch.sum(loss\_mask)

**4.BERT的三个Embedding为什么直接相加**

因为BERT模型需要同时考虑输入的token、位置和段落信息。在将它们相加之后，BERT模型可以同时获得这些信息。

**5.BERT的优缺点分别是什么？**

BERT模型的优点包括：

强大的语言**表示能力**：BERT能够学习到更丰富的语言表示，可以应对各种自然语言处理任务。

预训练模型的**通用性**：BERT在进行预训练后在有标注数据上进行微调，适用于各种任务和语言。

BERT模型缺点包括：

模型过于复杂：BERT拥有数亿个参数，推理时对计算及内存资源的要求较高。

学习时间较长： BERT需要预训练和微调，其训练时间相对较长，需要耗费大量的计算资源。

数据饥饿：BERT的预训练需要大规模无标注数据集，这对于资源有限的组织和个人来说可能是一个限制因素。

**6.你知道有哪些针对BERT的缺点做优化的模型？**

SpanBERT: SpanBERT是一种针对自然语言推理（NLI）任务的BERT模型改进，它通过对输入序列中的部分单词进行特殊标记，来使模型能够更好地理解上下文中的语义关系。

DistilBERT: DistilBERT是一种轻量化的BERT模型，通过剪枝和蒸馏技术来减少模型的大小和计算量，从而提高模型的训练速度和推理速度。

**7.BERT怎么用在生成模型中？**

使用BERT模型作为编码器，将输入序列转换为embedding，将embedding输入到decoder或者Diffusion中生成输出序列。该方法可以用于生成对话系统、机器翻译、文本摘要、生成文本图片对等任务。

GPT模型：使用BERT模型进行预训练，然后将预训练好的BERT模型作为初始参数，使用Transformer解码器进行微调，从而实现对文本生成任务的优化。GPT模型可以用于生成自然语言文本、文章摘要、问答系统等任务。