BERT（Bidirectional Encoder Representations from Transformers）来自谷歌人工智能语言研究人员发表的论文 BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding。它在各种 NLP 任务：包括机器问答（SQuAD v1.1）、自然语言推理（MNLI）等中展示最先进的结果，在机器学习社区引起了轰动。

BERT的出现，将NLP领域的预训练模型带入了一个新的纪元，其最重要的创新点在于训练策略的改变，将以往基于**自回归（Auto Regression,自左向右生成）**的训练策略转换为基于**去噪自编码(Denoising Auto Encoding)**的训练策略，即MLM任务。这使得词向量从先前只包含前文信息变为了可以学习到上下文的信息，虽然丢失了对自然语言**生成任务**的先天优势，但加强了词向量本身的特征。

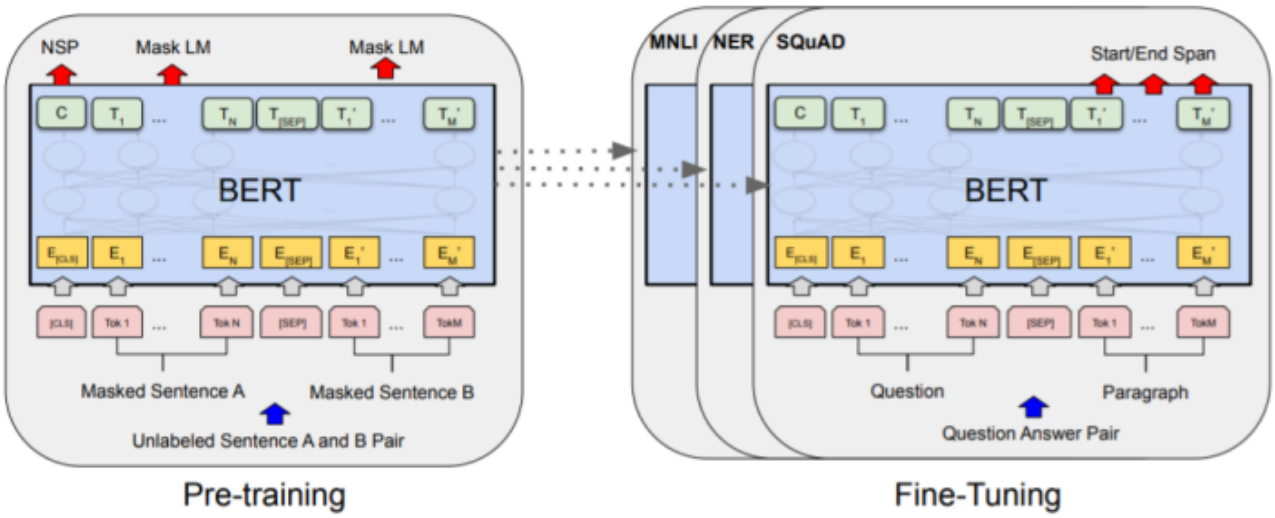
BERT 的关键技术创新是将流行的注意力模型 Transformer 的 双向训练应用于语言建模。论文的结果表明，与单向语言模型相比，双向训练的语言模型可以把握更多语言上下文的信息。Masked LM (MLM) 允许模型进行双向训练。

- Bidirection：BERT的整个模型结构是双向的。
- Encoder：是一种编码器，BERT只是用到了Transformer的Encoder部分。
- Representation：做词的表征。
- Transformer：Transformer是BERT的核心内部元素。

1.背景

先前，在计算机视觉领域，研究人员已经展示了迁移学习的价值——在已知任务上**预训练**神经网络模型，例如ImageNet，然后进行**微调**——使用训练好的神经网络作为新的特定目的模型。近年来，研究人员已经表明，类似的技术可以用于许多自然语言任务。

BERT就是先用Masked Language Model+Next Sentence Prediction两个任务做预训练，之后遇到新的任务时(如机器问答、NER)再微调：



2. BERT的原理

BERT 使用 Transformer，这是一种注意力机制，可以学习文本中单词（sub-word）之间的上下文关系。Transformer 包括两个独立的机制——一个读取文本输入的Encoder和一个为任务生成预测的Decoder。BERT只用了Encoder。

与顺序读取文本输入（从左到右/从右到左）的directional模型相反，Transformer 的Encoder一次读取整个单词序列。因此它被认为是双向(bi-directional)的，尽管更准确地说它是非定向的(non-directional)。这个特性允许模型根据单词的所有上下文来学习单词在上下文中的embedding。

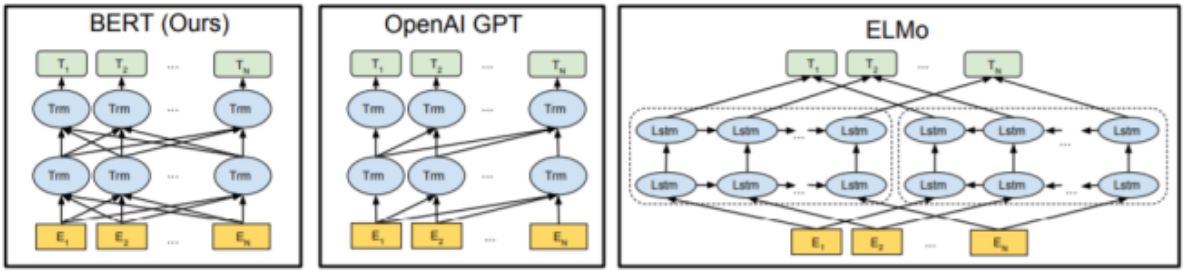


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT representations are jointly conditioned on both left and right context in all layers. In addition to the architecture differences, BERT and OpenAI GPT are fine-tuning approaches, while ELMo is a feature-based approach.

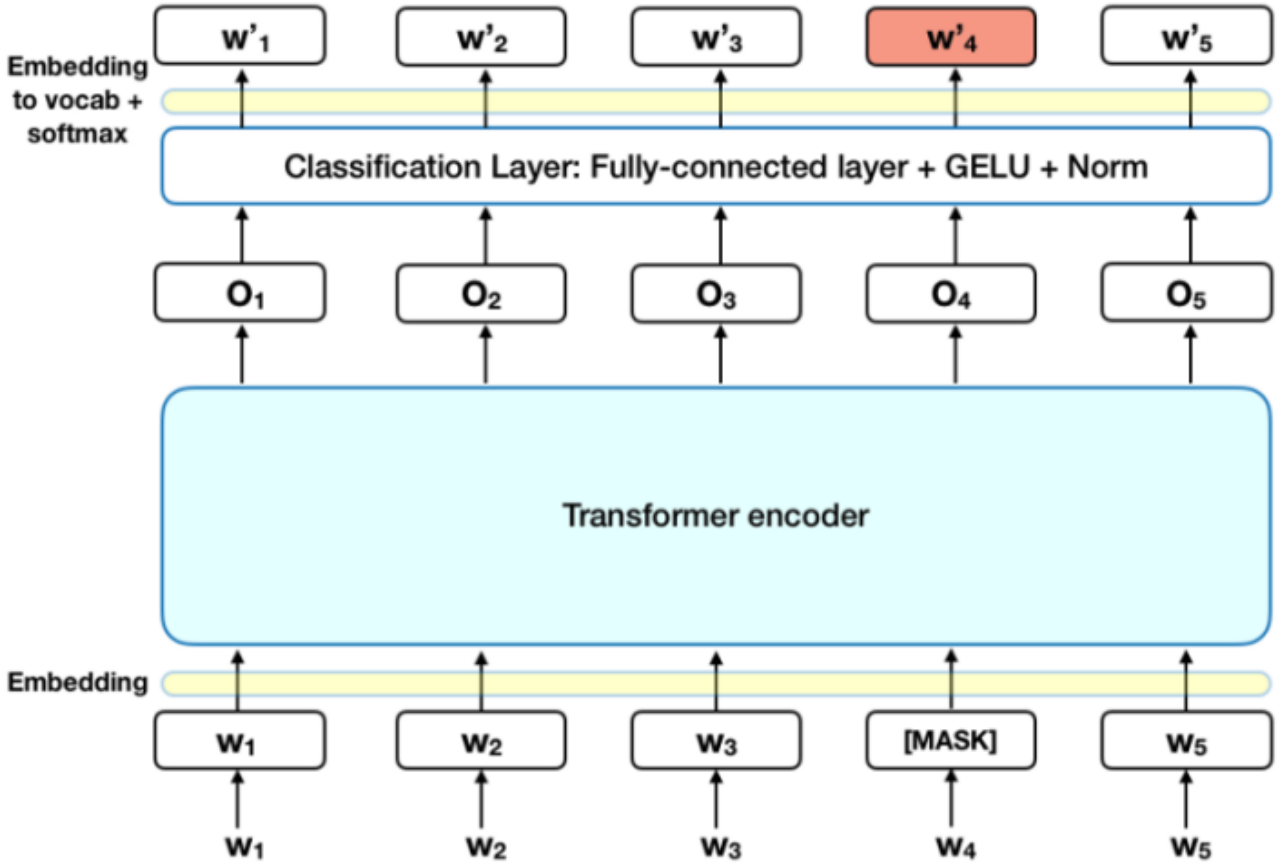
BERT是如何做预训练的两个任务：一是Masked Language Model（MLM）；二是Next Sentence Prediction（NSP）。在训练BERT的时候，这两个任务是**同时训练**的。所以，BERT的损失函数是把这两个任务的损失函数加起来的，是一个多任务训练。

2.1 Masked Language Model (MLM)

什么是Masked Language Model？它的灵感来源于完形填空。具体在BERT中，掩盖了15% 的Tokens。这掩盖了15%的Tokens又分为三种情况：

- 有80%的字符用“MASK”这个字符替换，如：My dog is hairy -> My dog is [MASK].
- 有10%的字符用另外的字符替换，如：My dog is hairy -> My dog is apple
- 有10%的字符是保持不动，如: My dog is hairy -> My dog is hairy.

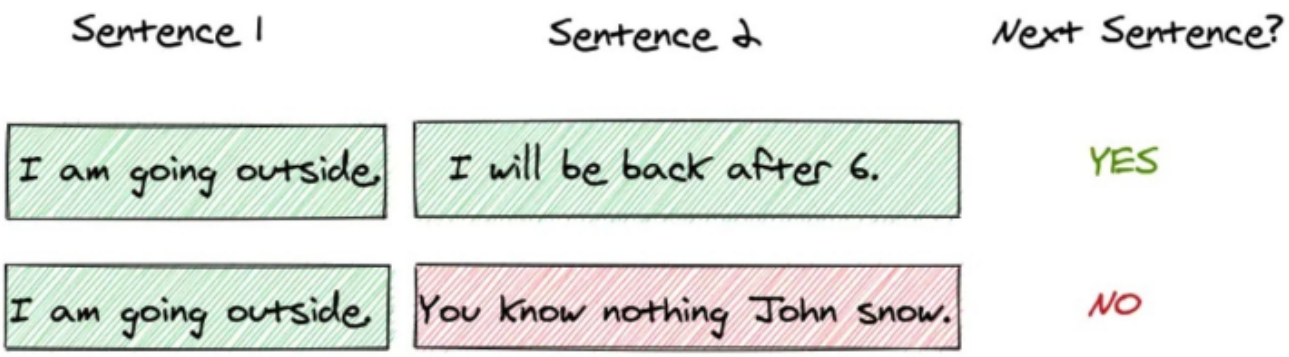
让模型去预测/恢复被掩盖的那些词语。最后在计算损失时，只计算被掩盖的这些Tokens(也就是掩盖的那15%的Tokens)。



2.2 Next Sentence Prediction

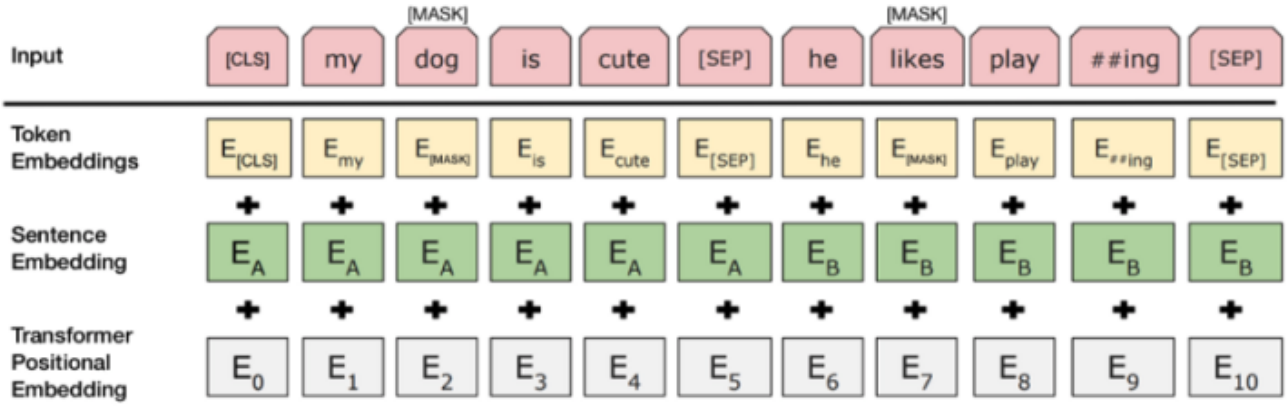
Next Sentence Prediction是更关注于两个句子之间的关系。与Masked Language Model任务相比，Next Sentence Prediction更简单些。

在 BERT 训练过程中，模型的输入是一对句子<sentence1,sentence2>，并学习预测sentence2是否是原始文档中的sentence1的后续句子。在训练期间，50% 的输入是一对连续句子，而另外 50% 的输入是从语料库中随机选择的不连续句子。



为了帮助模型区分训练中的两个句子是否是顺序的，输入在进入模型之前按以下方式处理：

- 在第一个句子的开头插入一个 [CLS] 标记，在每个句子的末尾插入一个 [SEP] 标记。
- 词语的embedding中加入表示句子 A 或句子 B 的句子embedding（下图中的Sentence Embedding）。句子embedding其实就是Vocabulary大小为2的embedding。
- 加入类似Transformer中的Positional Embedding

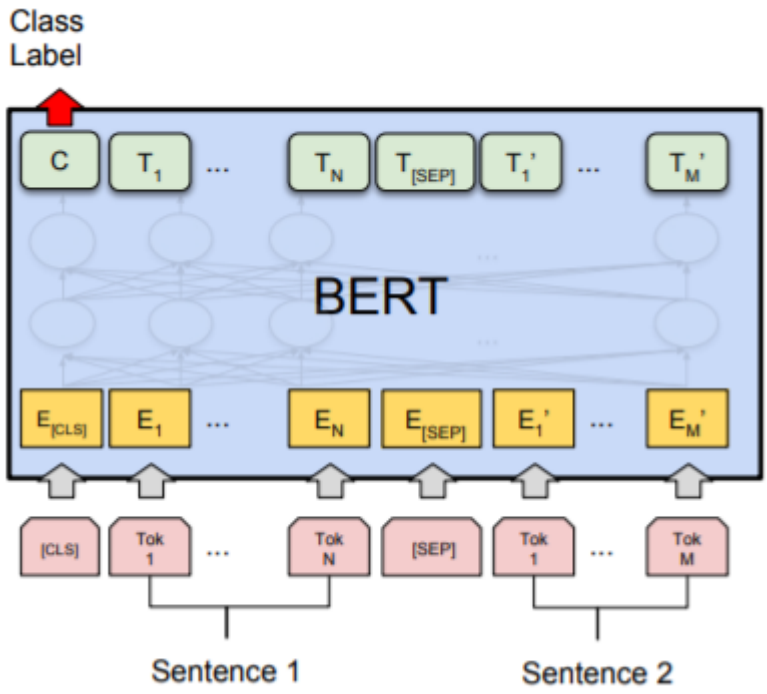


特殊字符介绍：

- [CLS]，全称是Classification Token（CLS），是用来做一些分类任务。[CLS] token为什么会放在第一位？因为本身BERT是并行结构, [CLS]放在尾部也可以，放在中间也可以。放在第一个应该是比较方便。
- [SEP]，全称是Special Token（SEP），是用来区分两个句子的，因为通常在train BERT的时候会输入两个句子。从上面图片中，可以看出SEP是区分两个句子的token。

为了预测第二个句子是否确实是第一个句子的后续句子，执行以下步骤：

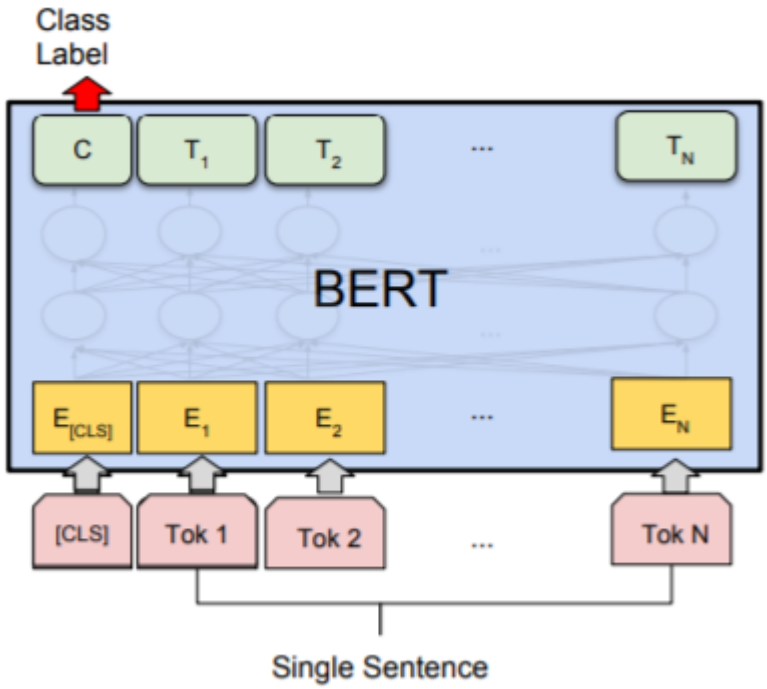
- 整个输入序列的embedding被送入Transformer 模型
- [CLS]对应的输出经过简单MLP分类层变成2*1向量([isNext,IsnotNext])
- 用softmax计算IsNext的概率



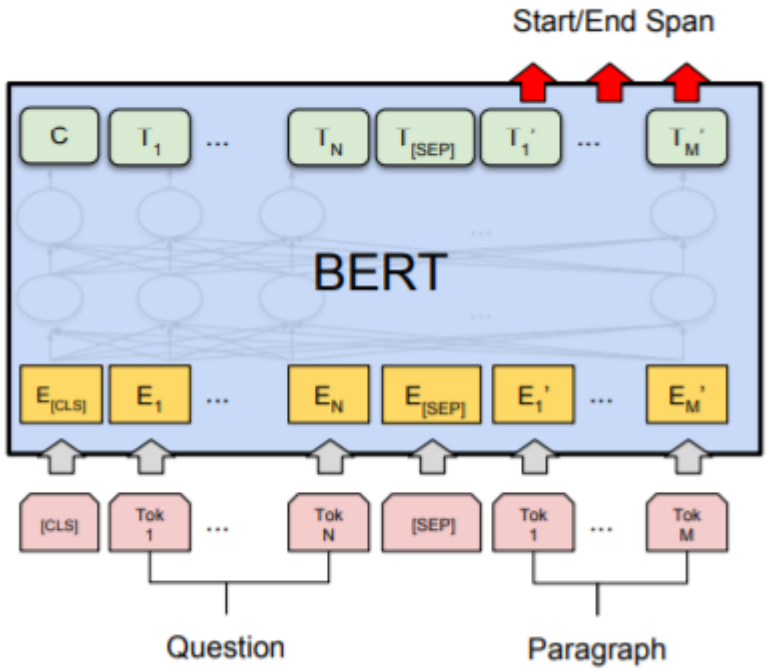
3. 如何用BERT做Fine-tuning

BERT 经过微小的改造（增加一个小小的层），就可以用于各种各样的语言任务。

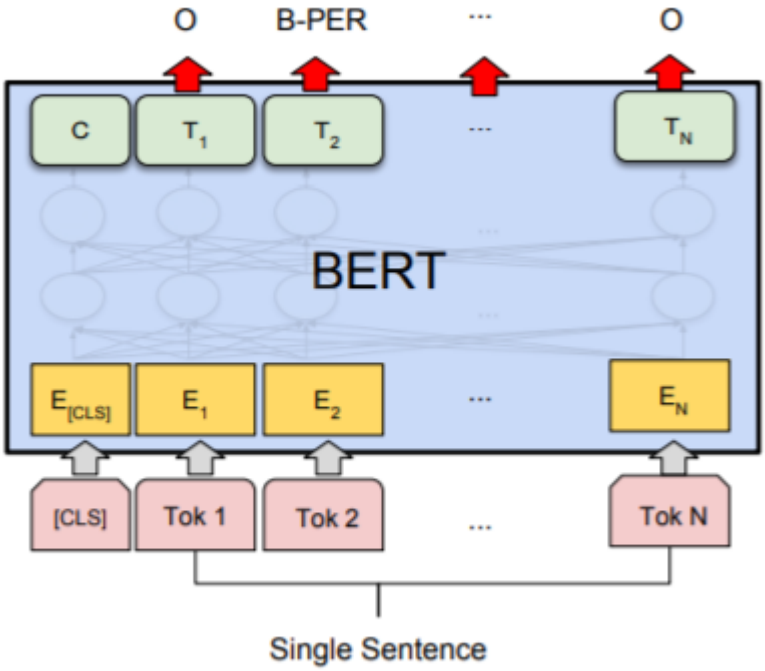
（1）与 Next Sentence Prediction类似，通过在 [CLS] 标记的 Transformer 输出顶部添加分类层，完成诸如情感分析之类的分类任务：



（2）在问答任务（例如 SQuAD v1.1）中，会收到一个关于文本序列的问题，并需要在序列中标记答案。使用 BERT，可以通过学习**标记答案开始和结束的两个额外向量**来训练问答模型。



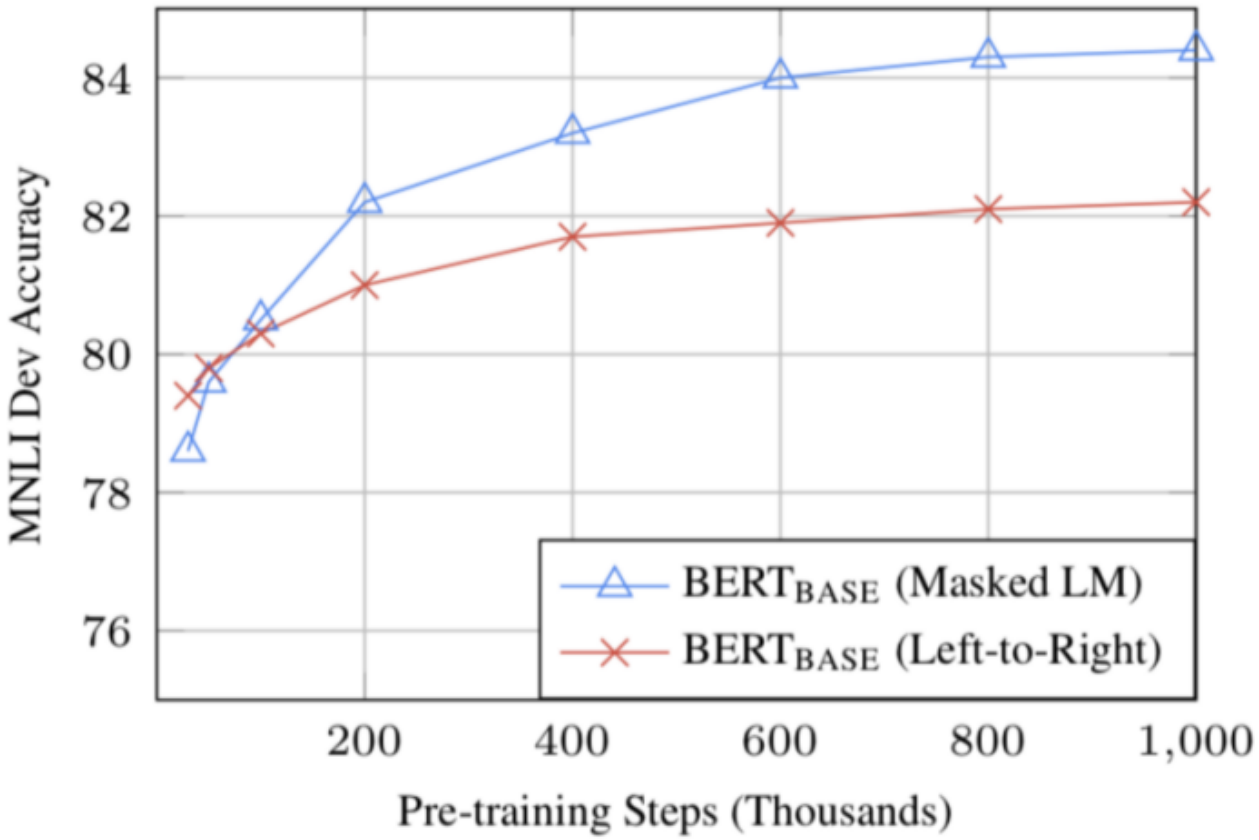
（3）在命名实体识别 (NER) 中，接收文本序列，并需要标记文本中出现的各种类型的实体（人、组织、日期等）。使用 BERT，可以通过将每个标记的输出向量输入到预测 NER 标签的分类层来训练 NER 模型。



4. Takeaways

(1) BERT官方提供了两个版本的BERT模型。一个是BERT的BASE版，另一个是BERT的LARGE版。BERT的BASE版有12层的Transformer，隐藏层Embedding的维度是768，head是12个，参数总数大概是一亿一千万。BERT的LARGE版有24层的Transformer，隐藏层Embedding的维度是1024，head是16个，参数总数大概是三亿四千万。

(2)BERT 的bi-directional方法 (MLM) 的收敛速度比从左到右的directional方法慢（因为每批中**只预测了 15% 的单词**，而自回归语言模型每个样本都有**全部**的单词参与训练），但经过少量预训练步骤后，双向训练仍然优于从左到右的单向训练。



参考资料：

https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270towardsdatascience.com