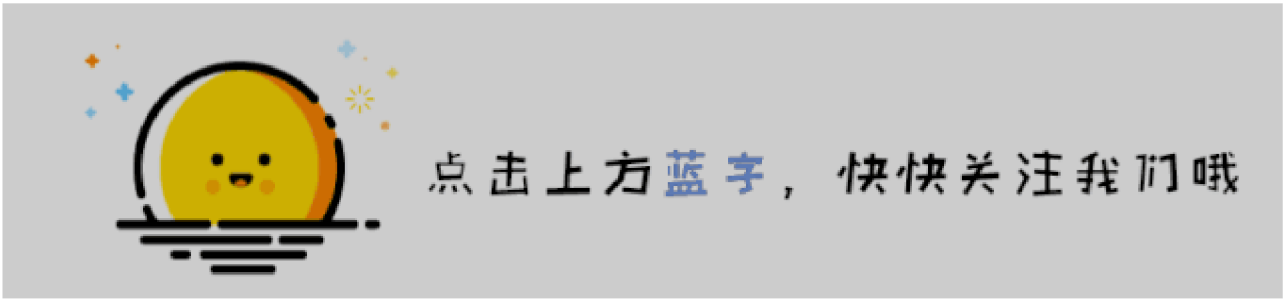
[预训练语言模型专题] BERT，开启NLP新时代的王者

原创 管扬 朴素人工智能

来自专辑

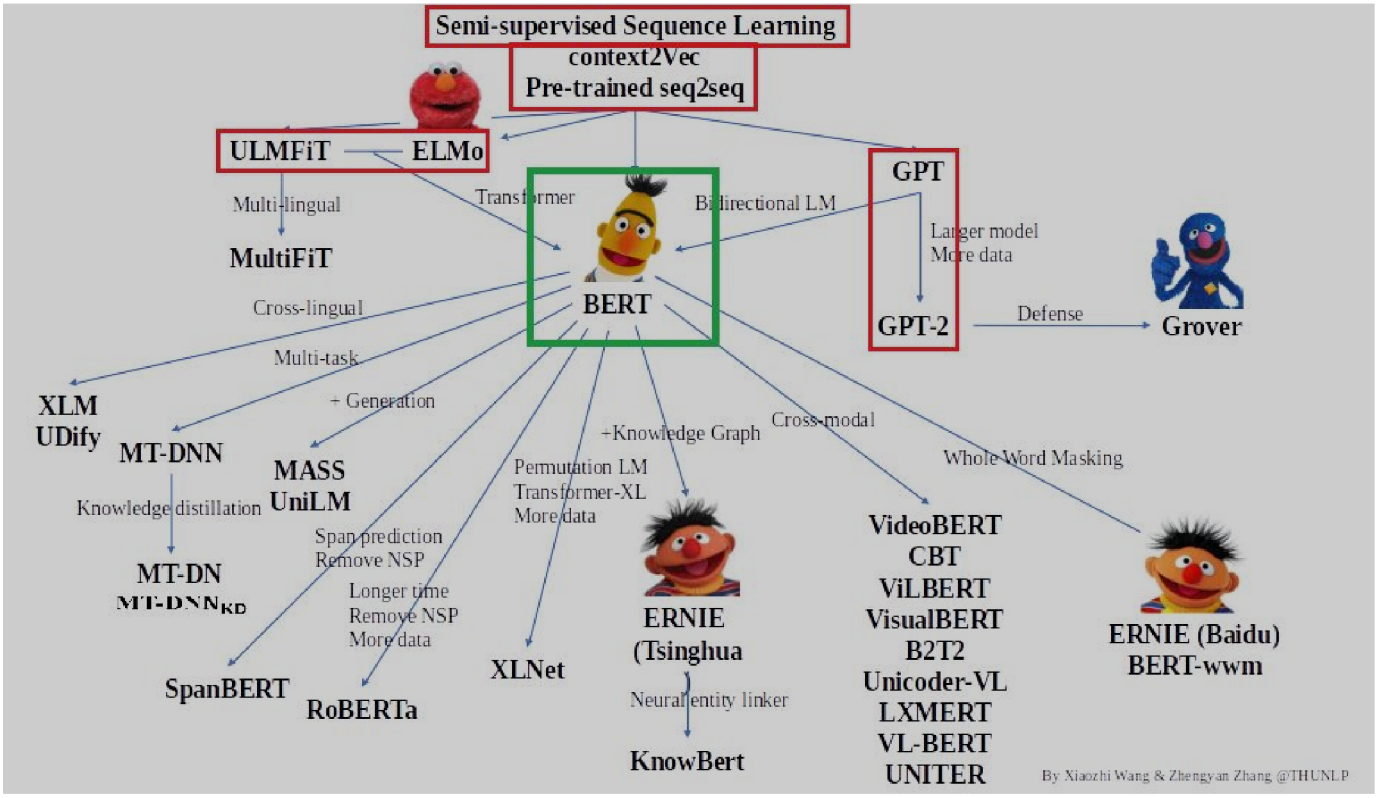
预训练语言模型



本文为预训练语言模型专题系列第五篇

前期回顾：[萌芽时代]、[风起云涌]、[文本分类通用技巧] 、[GPT家族]

感谢清华大学自然语言处理实验室对预训练语言模型架构的梳理，我们将沿此脉络前行，探索预训练语言模型的前沿技术，红色框为前期脚印，绿色框为本期介绍，欢迎大家留言讨论交流。



BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (2018)

本期将要介绍的就是在NLP领域无人不知，无人不晓的预训练语言模型BERT了，由Google AI在2018年底推出，刚出现就刷新了一大批榜单，甚至在一些任务上**超过了人类的表现**，令人惊艳。谷歌团队成员Thang Luong在推特上表示，BERT模型开启了NLP的新时代。就其效果、易用性、通用性各方面来说，在当时不愧称为预训练语言模型的王者，压服众多的模型。让我们花十分钟一起，跟着论文来体会BERT的设计思路和重点。

文章在一开始概括了当时的两种不同的预训练语言模型的策略，feature-based 策略及 fine-tuning 策略。

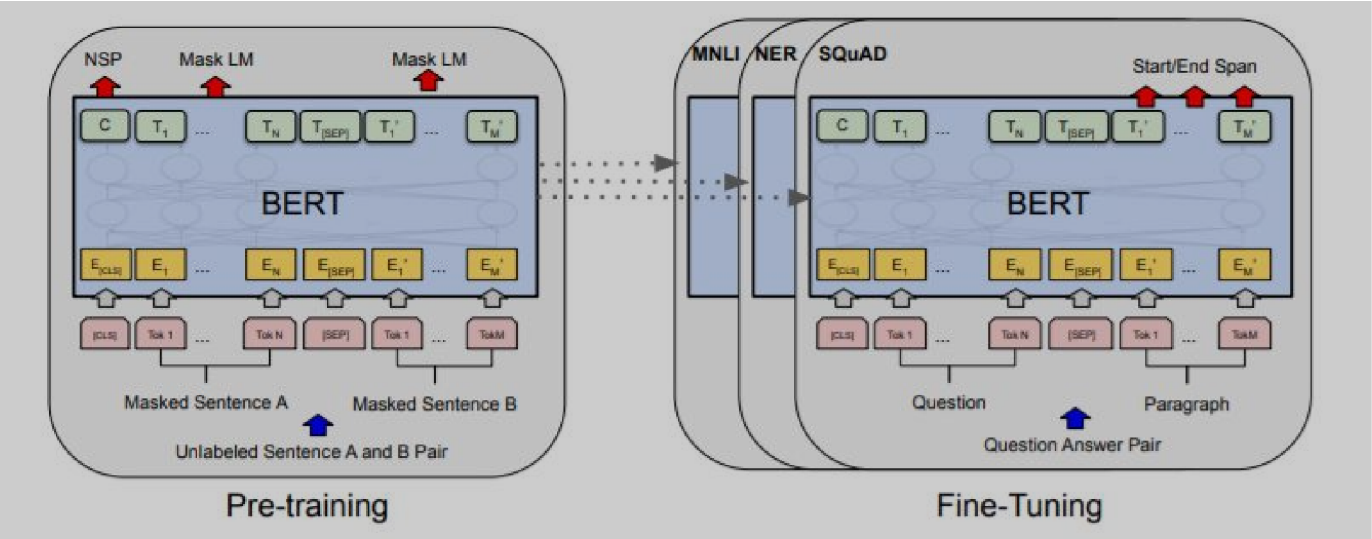
1. **feature-based**策略的代表模型为ELMo，它把预训练得到的“向量表示”作为训练下游任务的额外特征。训练下游任务时使用新的任务相关模型，并基于得到的特征来进行进一步的训练。
2. **fine-tuning**策略的代表模型为GPT，它则是在训练语言模型后，只改变极少的任务相关的层与参数，直接对下游任务训练整个原来的语言模型。

BERT使用的是后者，因为这种策略需要改变的参数量较少，迁移也较为简单。同时他们指出，现在限制这种策略性能的主要问题是。如GPT这种模型，它预训练时使用了标准语言模型的目标，导致它只能是单向的。在Transformer层中，每个token在self attention 时都只能关注其之前的token，会严重损害在一些任务上如问答上的效果，在之前的不少论文中都佐证了这个观点。所以，使模型能够学习一个文本**双向的信息**是非常关键的一点。

BERT模型的几大核心贡献：

1. BERT揭示了语言模型的**深层双向学习能力**在任务中的重要性，特别是相比于同样在fine-tuning范畴内使用单向生成式训练的GPT以及浅层的双向独立训练并 concat 的ELMo，BERT的训练方法都有了很大的进步，BERT是通过改进训练目标来实现深层双向的语言模型训练，待会会单独介绍
2. BERT再次论证了**fine-tuning**的策略是可以有很强大的效果的，而且再也不需要为特定的任务进行繁重的结构设计。BERT也是使用fine-tuning策略的模型中第一个无论在句级别或在词级别都获得了state-of-art效果，胜过了不少专为相关任务设计的模型。
3. BERT在11个NLP任务上获得了state-of-art的效果，在SQuAD v1.1 问答任务上**超过人类水平**。

BERT如此受大家推崇的原因之一当然是其强劲的性能，但另外突出的是其易用性和通用性。BERT的预训练和下游特定任务的训练，在模型上的差别仅仅是顶层的output layer ，而且可以在很多任务上通用。



BERT的最大创新是在预训练的时候使用了两个非监督任务

1. **Masked LM** (MLM)

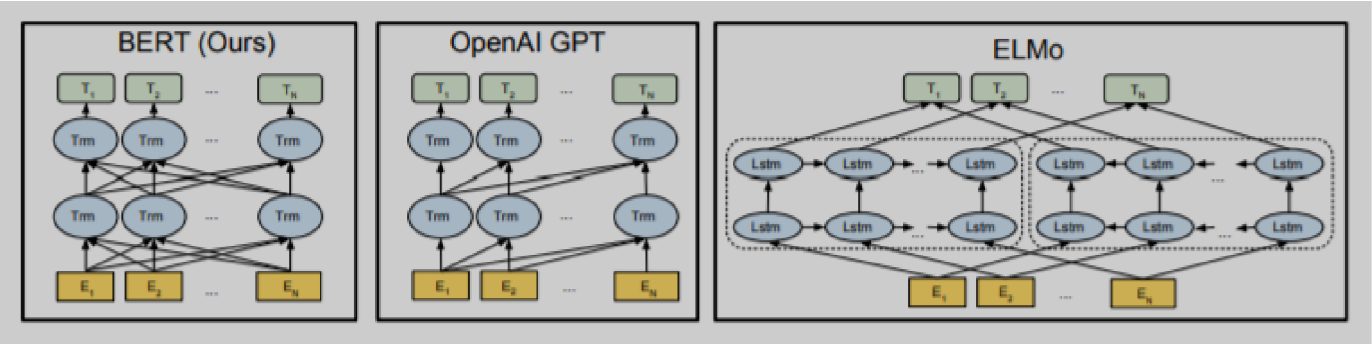
在前面的介绍也讲到，作者认为无论是单向的生成式语言模型，还是独立的left-to-right和right-to-left的进行拼接都不如真正的深层双向联合训练。但以标准的语言模型目标，没办法实现双向的训练，因为模型在预测某个单词时，会间接地在多层的上下文中看见“自己”，导致**泄露**。

BERT提供的解决方案就是Mask LM 任务，它会随机mask掉一定比例的token，让它在训练的时候不在输入中出现，并把它们作为目标来训练，这样就可以防止泄露，mask的方式是把token替换成一个固定的token [MASK]。在实际使用的过程中，这会带来一个问题，因为MLM任务是语言模型的训练任务，也就是说[MASK] 这种token只会在语言模型训练时有，在下游模型的fine-tuning时是不会出现的，这就会导致预训练和fine-tuning的数据分布不匹配。为了弥补这个问题，这15%应该被mask掉的token有80%的可能被替换成[MASK]，有10%的可能被替换成另外一个随机的token，另有10%的可能会维持原样不变。这样做，可以让Transformer的encoder无法去知道哪个token是要被预测的，也不知道哪个词被替换成随机词了，使它不得不对每一个token都形成一个较好的向量表示，没法取巧。

2. **Next Sentence Prediction** (NSP)

> 很多任务，包括问答、自然语言推断等是基于理解**两句句子之间关系**的，不能直接被语言模型所建模，所以BERT还有另外一个二分类任务NSP来捕捉句子间的关系。在构造这个任务的数据集时，会有50%的概率，提供正样本，即某句句子和其下一句句子的组合，50%的概率在语料中选择任意一句句子构成负样本。这个任务相较MLM来说还是相当简单的。

BERT模型细节：



我一直认为BERT论文中的这张图画得非常好，将三种模型的结构描绘得十分清楚，我们从右到左再来重温一下，如果对细节想要进一步了解，建议回顾前几期的推送。

- ELMo模型的核心组件是LSTM。最下方的Embedding层为字向量；中间是两层LSTM，分别有独立的left-to-right 和 right-to-left 的双向网络；双向LSTM的输出在最上方连接，形成包含上下文语义的向量表示。
- GPT模型的核心组件首次使用了Transformer。最下方的Embedding层为token embedding 与position embedding 相加，token embedding的vocab为BPE算法所得；中间为12层的Transformer，语言模型目标为标准的单向条件概率，没有双向的语义能力。
- BERT模型的核心组件是Transformer。最下方的Embedding为token embedding、segment embedding和position embedding 相加。token embedding的vocab为30000个左右词的Wordpiece embedding；中间的Transformer层取决于两个模型的尺度；因为MLM任务，所以BERT能够捕捉双向的语义特征。
  - BERT-base 12层，hidden size 768，attention heads 12，总参数110M。base的所有尺度都是和GPT做对标的
  - BERT-large 24层，hidden size 1024，attention haads 16，总参数量340M。

BERT的输入也与GPT类似都用了[CLS]和[SEP]，相比之下在预训练和finetune都做了规范化和处理，以应对不同的任务。句子开头的token为[CLS]，结尾的token为[SEP]。如果输入仅有一句话，那规范化后的tokens是[CLS] [Sentence1] [SEP]，如果为两句话，那么规范后的tokens是 [CLS] [Sentence1] [SEP] [Sentence2] [SEP] 。另外，BERT模型还需要输入segment_id，以标识token的每一个位置是属于第一句话还是第二句话的，第一句话的位置上segment_id都是0，第二句话的位置都是1。具体的细节，下一期我打算结合hugging face的transformers的代码来进行进一步的分享。

BERT预训练所用的数据更大了，包含BookCorpus（800M words）和English Wikipedia（2500M words）

对比实验

接下来，文章对BERT模型中对性能产生影响的各个因素进行了对比试验。

首先是预训练任务的影响。LTR指的是Left-to-Right，可以看出最大的收益来源于Transformer替代了BiLSTM，其次是MLM任务带来的双向深层训练，再其次是NSP任务带来的收益。

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

各预训练任务带来的影响

接着是模型尺度的影响，可以很明显的在图中看出，总体来说是**越大的模型**会获得越好的效果。

Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

模型尺度带来的影响

最后，如果把BERT当作feature-based模型来用，不同层的向量表征所影响的效果，可以看到最好的是**最后四层concat起来**的结果，其实我们在[文本分类训练技巧] 一文中，专门就这个进行过讨论，有兴趣可以移驾一看。

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

不同层向量选择对feature-base模型带来的影响

不得不说，BERT的来到在2018年底给NLP的格局带来了巨大的变化，它让人们对NLP的前景充满了信心和期待，机器在NLP领域真的有了超越人类的可能，令人振奋。但BERT预训练的代价可不小，文中写道，模型batch size为256：(256 sequences * 512 tokens = 128,000 tokens/batch)，总共大概在33亿左右的单词上训练了40个epoch。BERT-base在4 Cloud TPUs (16 TPU chips total）上，BERT-large在 16 Cloud TPUs (64 TPU chips total)，都训练了大概4天左右才训练完毕。

这么多算力的消耗说实话也让人大吃一惊，换成普通的GPU时间简直是令人绝望的数字。不过下游的任务的预训练和特定任务训练就很快了。对于一般的开发者来说，只要下载在通用语料上预训练的模型，接着进行时间上可接受的下游训练即可，这也就是预训练语言模型给我们带来的魅力！

未完待续

本期的论文就给大家分享到这里，感谢大家的阅读和支持，下期我们大概率会给大家带来实打实的huggingface transformers上相关代码分析，敬请大家期待！

欢迎关注朴素人工智能，这里有很多最新最热的论文阅读分享，有问题或建议可以在公众号下留言。

参考资料

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

   https://arxiv.org/pdf/1810.04805.pdf

推荐阅读

- 十分钟了解文本分类通用技巧
- 一次实体识别和实体消岐的积极尝试
- Kaggle TensorFlow 2.0 Question Answering 16名复盘
- [预训练语言模型的前世今生] 风起云涌
- LaserTagger: 文本生成任务的序列标注解决方案