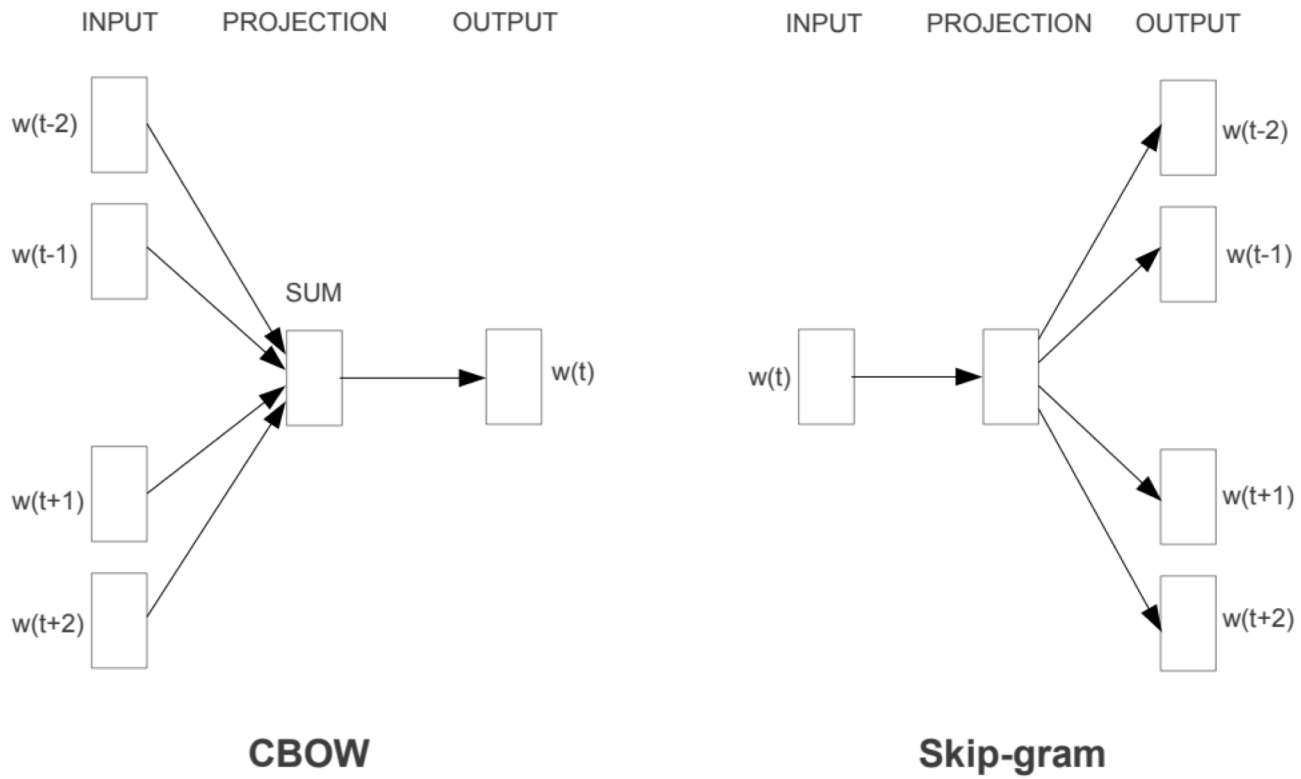
# BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

**Jacob Devlin** **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

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

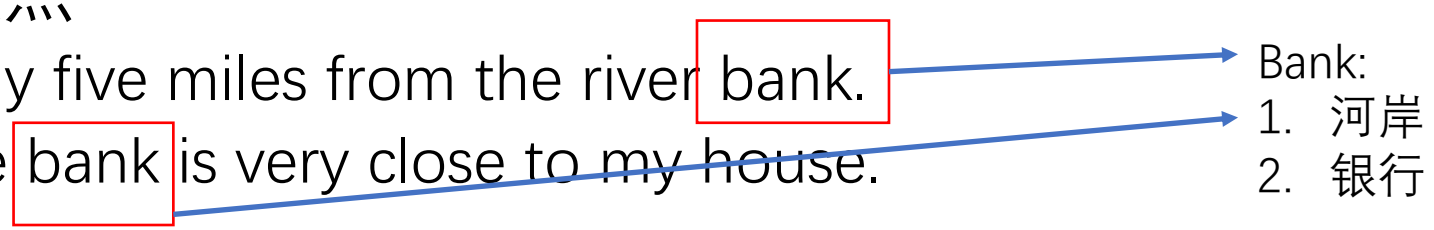
# 神经网络语言模型

- Word2Vec：如果上下文是相似的，语义也是相似的。



# 神经网络语言模型

- Word2Vec缺点
  - Sent1：Only five miles from the river bank.
  - Sent2：The bank is very close to my house.




- 只能得到静态的Word Embedding
- 对于一词多义表示的不够好

# 神经网络语言模型

- 动态Word Embedding的NNLM

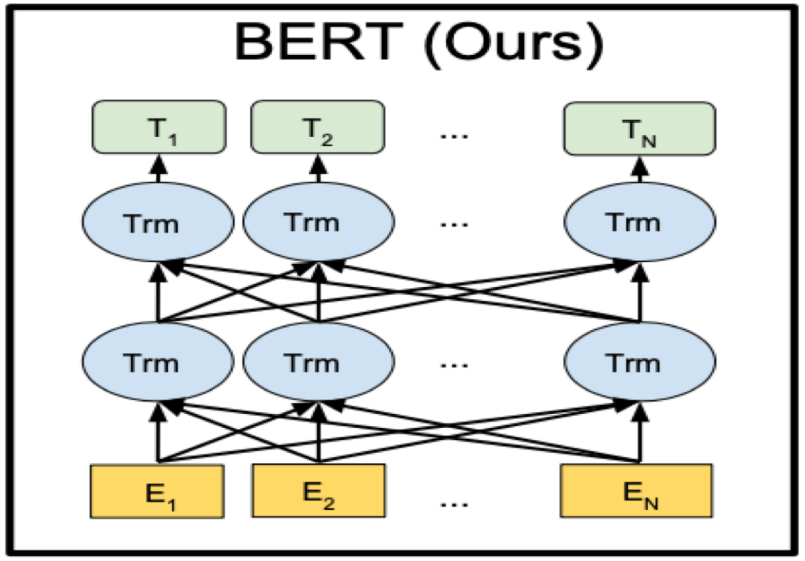
1. ELMO（论文： Deep contextualized word representations ）
2. GPT（论文： Improving Language Understanding with Unsupervised Learning）
3. BERT

# BERT

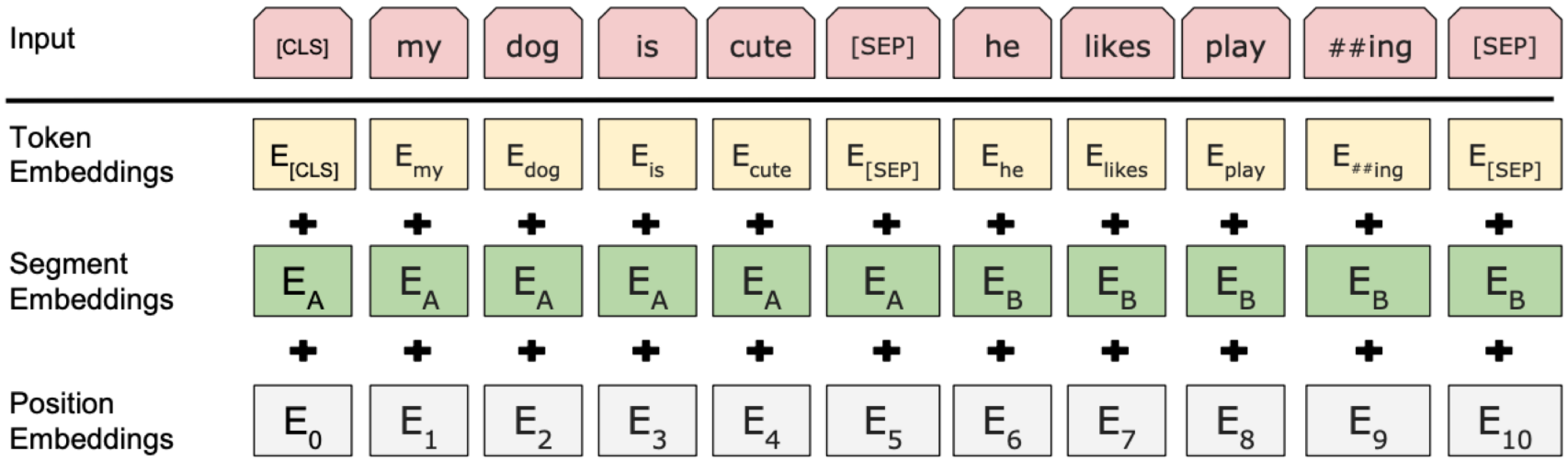
- **B**idirectional **E**ncoder **R**epresentations from **T**ransformers
- 2018年由谷歌提出
- 刷新了自然语言处理的11项记录
- 对NLP来说有非常重要的意义（里程碑）

# BERT结构

- 网络结构：窄而深
  - $BERT_{BASE}$：12层
  - $BERT_{LARGE}$：24层
- 每一层都是一个 “Transformer块”



# BERT输入




每个位置的输入均有三部分构成，三个向量加和作为当前词的输入

1. Token级别的向量
2. Segment级别的向量
3. 位置向量

*第一个位置是[CLS]标记，句子结尾是[SEP]标记

# BERT预训练过程

- Masked Language Model
  - 随机mask每一个句子中15%的词，用其上下文来做预测，例如：my dog is hairy → my dog is [MASK]

- Next Sentence Prediction
  - 选择一些句子对A与B，其中50%的数据B是A的下一条句子，剩余50%的数据B是语料库中随机选择的，预测两个句子是否是上下文关系。

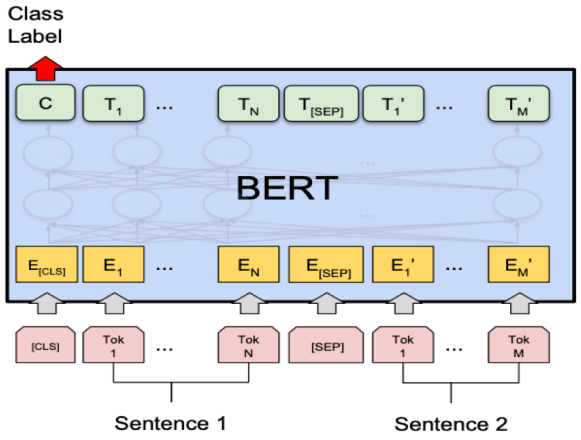
$$\text{Input} = \texttt{[CLS] the man went to [MASK] store [SEP]}$$
$$\texttt{he bought a gallon [MASK] milk [SEP]}$$
$$\text{Label} = \texttt{IsNext}$$

# BERT在下游任务的应用方式

- Fine-tuning（推荐）
  - 根据相应的任务对BERT的输出结构进行改造
  - **加载预训练的模型参数**
  - 在相应的任务上继续训练 → BERT模型参数会随着下游任务的训练而发生变化
- 使用词向量
  - 获取到BERT输出的词向量 → 只获取了输出，BERT模型参数不变
  - 作为其他网络结构的输入
  - 训练下游任务的模型

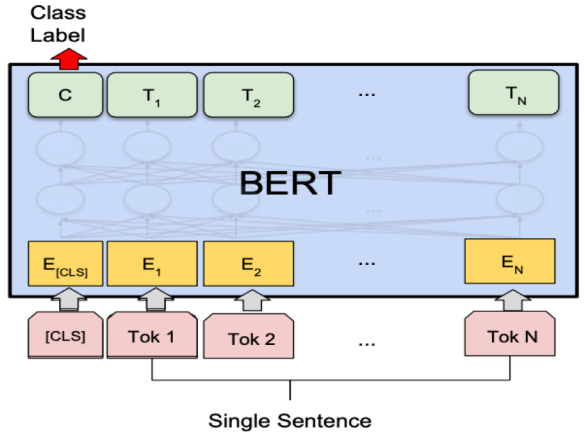
# 不同任务的Fine-tuning

a.句子对分类任务



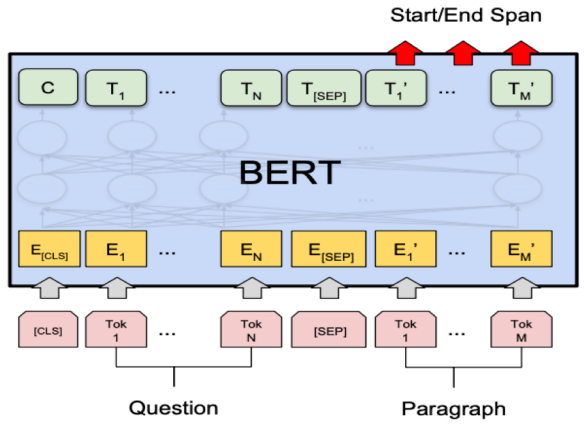

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

b.单句分类任务



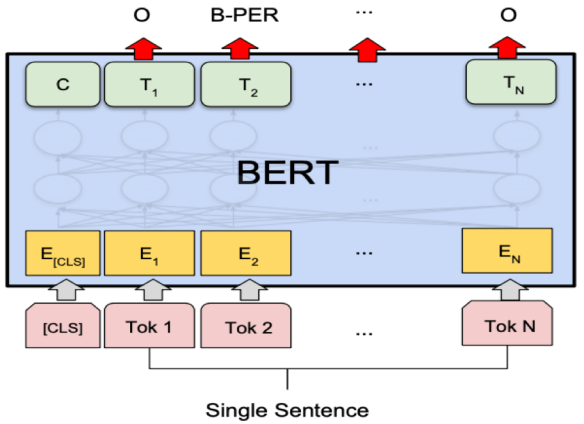

(b) Single Sentence Classification Tasks: SST-2, CoLA

c.机器问答任务




(c) Question Answering Tasks: SQuAD v1.1

d.序列标注任务




(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

# BERT优点

- 充分利用了大量的无监督语料，学习到了一些语言学知识
- 利用Transformer作为特征提取器
- 动态词向量更恰当的表示语义信息
- 效果非常好

# BERT的实现

- TensorFlow版本（ https://github.com/google-research/bert）
- Pytorch版本（ https://github.com/huggingface/transformers ）

# BERT的学习参考

- http://www.52nlp.cn/tag/bert%E8%A7%A3%E8%AF%BB