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

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language

- ELMo = **E**mbeddings from **L**anguage **Mo**dels
- GPT = **G**enerative **P**re-**T**raining
- BERT = **B**idirectional **E**ncoder **R**epresentations from **T**ransformers

We introduce a new language representation model called **BERT,** which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers

芝麻街系列文章

(ERNIE, Grover, Big Bird, Kermit, RoBERTa, Rosita, …)

Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial taskspecific architecture modifications.

BERT与ELMo和GPT的区别：

GPT考虑单向的，它用左边的上下文信息去预测未来。BERT用了左侧和右侧信息，是双向的。ELMo用的是基于RNN的框架，而BERT用的是Transformer。所以ELMo在用到一些下游任务时，需要对架构做调整，而BERT仅改最上层即可。

语言模型

语言模型通俗点讲就是计算一个句子的概率。通常来说指的是前向语言模型

对于语言序列 w1,w2,⋯ ,wn，语言模型就是计算该序列的概率，即 P(w1,w2,⋯ ,wn)。

给定一句由 n 个词组成的句子 W=w1,w2,⋯ ,wn，计算这个句子的概率 P(w1,w2,⋯ ,wn)，或者计算根据上文计算下一个词的概率 P(wn|w1,w2,⋯ ,wn−1)。

Conditional probability:

$p(w \mid w_1, w_2), \forall w \in V$

$$p(w_1, w_2, w_3, \ldots, w_n) = p(w_1)p(w_2 \mid w_1)p(w_3 \mid w_1, w_2) \times \cdots \times p(w_n \mid w_1, w_2, \ldots, w_{n-1})$$

Sentence: “the cat sat on the mat”

$$P(\text{the cat sat on the mat}) = P(\text{the}) * P(\text{cat}|\text{the}) * P(\text{sat}|\text{the cat}) * P(\text{on}|\text{the cat sat}) * P(\text{the}|\text{the cat sat on}) * P(\text{mat}|\text{the cat sat on the})$$

语言模型随处可见


Google
how is the weather in new
how is the weather in new york
how is the weather in new zealand
how is the weather in new orleans
how is the weather in new jersey
how is the weather in new orleans in february
how is the weather in new york in march
how is the weather in new orleans in january
how is the weather in new mexico
how is the weather in new york in february
how is the weather in new orleans in december
Google Search
I'm Feeling Lucky
Report inappropriate predictions


New Message
Cancel
To:
[无标题]
Language models are the
best
most
same

马尔可夫假设

- 使用最近的过去预测下一个单词

- 1st order

$$P(\text{mat}|\text{the cat sat on the}) \approx P(\text{mat}|\text{the})$$

- 2nd order

$$P(\text{mat}|\text{the cat sat on the}) \approx P(\text{mat}|\text{on the})$$

Andrey Markov

K阶马尔可夫链

Consider only the last *k* words (or less) for context

$$P(w_i \mid w_1 w_2 \ldots w_{i-1}) \approx P(w_i \mid w_{i-k} \ldots w_{i-1})$$

which implies the probability of a sequence is:

$$P(w_1 w_2 \ldots w_n) \approx \prod_i P(w_i \mid w_{i-k} \ldots w_{i-1})$$

(assume $w_j = \phi \quad \forall j < 0$)

后向语言模型

句子：$t_1, t_2, t_3, \ldots, t_n$

$$p(t_1,t_2,\ldots t_n) = p(t_n)\, p(t_{n-1}|t_n)\, p(t_{n-2}|t_n,t_{n-1}) \ldots\ldots p(t_1|t_n,t_{n-1},\ldots t_2)$$

$$p(t_1,t_2,\ldots t_n) = \prod_{k=n}^{1} p(t_k|t_n, t_{n-1}, \ldots, t_{k+1})$$

NLP发展历程

年份	2013 年	2014 年	2015 年	2016 年	2017 年
技术	word2vec	GloVe	LSTM/ Attention	Self-Attention	Transformer

年份	2018 年	2019 年	2020 年
技术	GPT/ELMo/ BERT/GNN	XLNet/BoBERTa/ GPT-2/ERNIE/T5	GPT-3/ELECTRA/ ALBERT

Limitations

- 词语和向量之间的一一对应关系

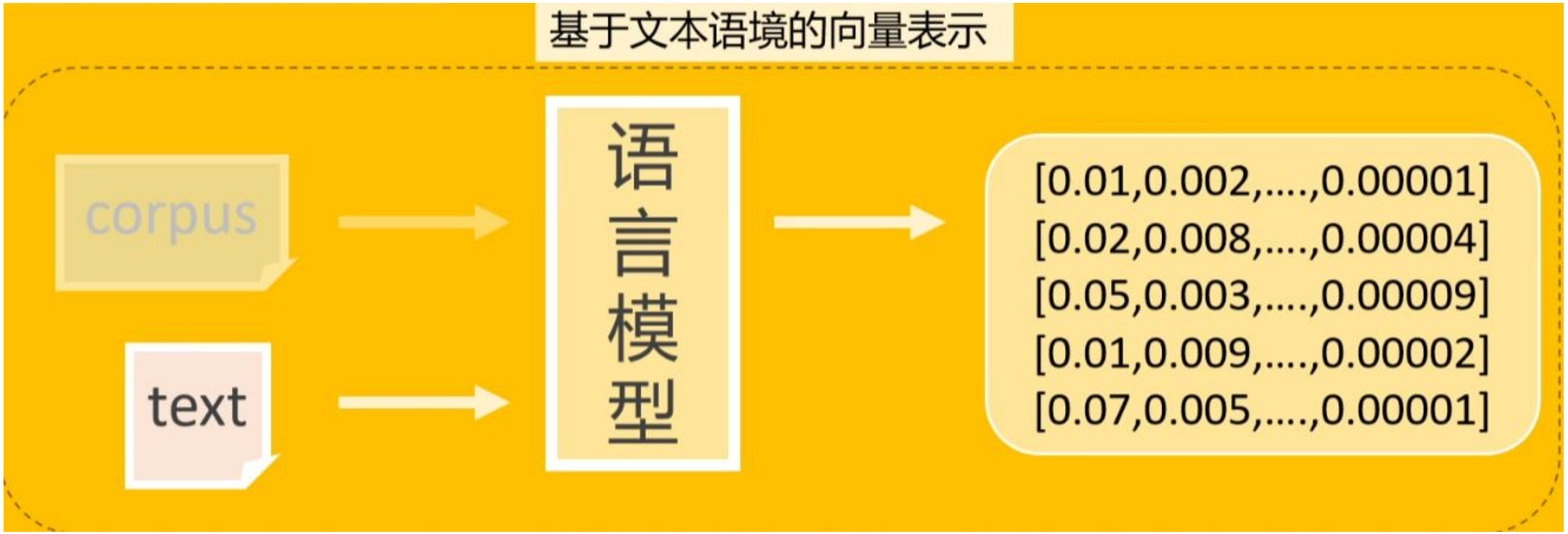
$$v(\text{play}) = \begin{pmatrix} -0.224 \\ 0.130 \\ -0.290 \\ 0.276 \end{pmatrix}$$

- 复杂的词语使用特征：句法和语义

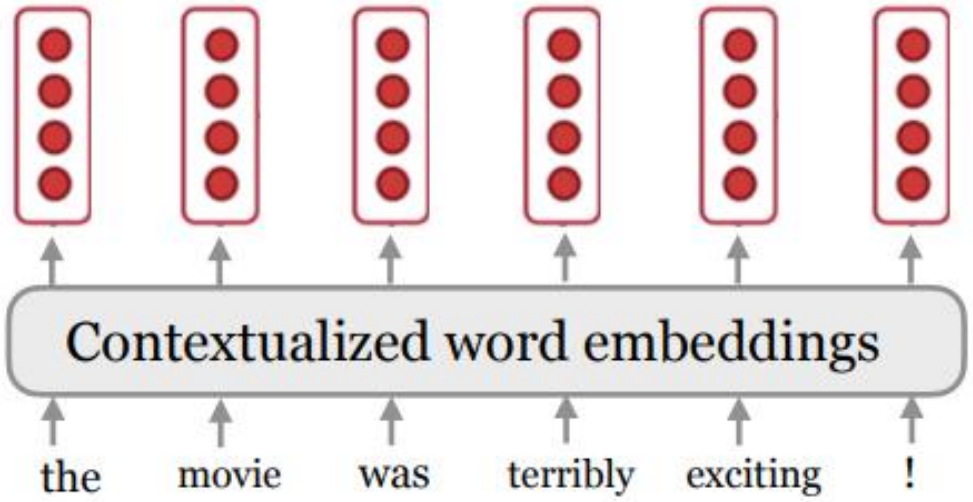
- 多义词，例如，bank,mouse

mouse^1 : .... a *mouse* controlling a computer system in 1968.
mouse^2 : .... a quiet animal like a *mouse*
bank^1 : ...a *bank* can hold the investments in a custodial account ...
bank^2 : ...as agriculture burgeons on the east *bank*, the river ...

文本语境词嵌入

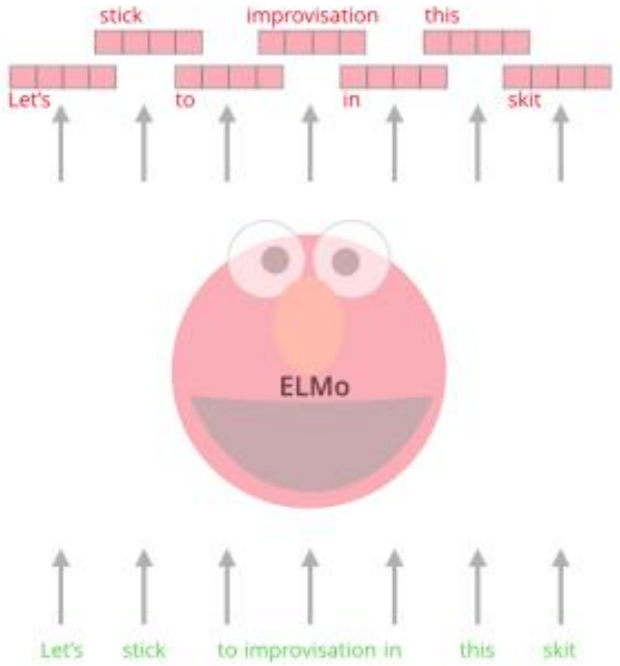

基于文本语境的向量表示
corpus
text
语言模型
[0.01,0.002,….,0.00001]
[0.02,0.008,….,0.00004]
[0.05,0.003,….,0.00009]
[0.01,0.009,….,0.00002]
[0.07,0.005,….,0.00001]

文本语境词嵌入



$$f : (w_1, w_2, \ldots, w_n) \longrightarrow \mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$$

ELMo Embeddings



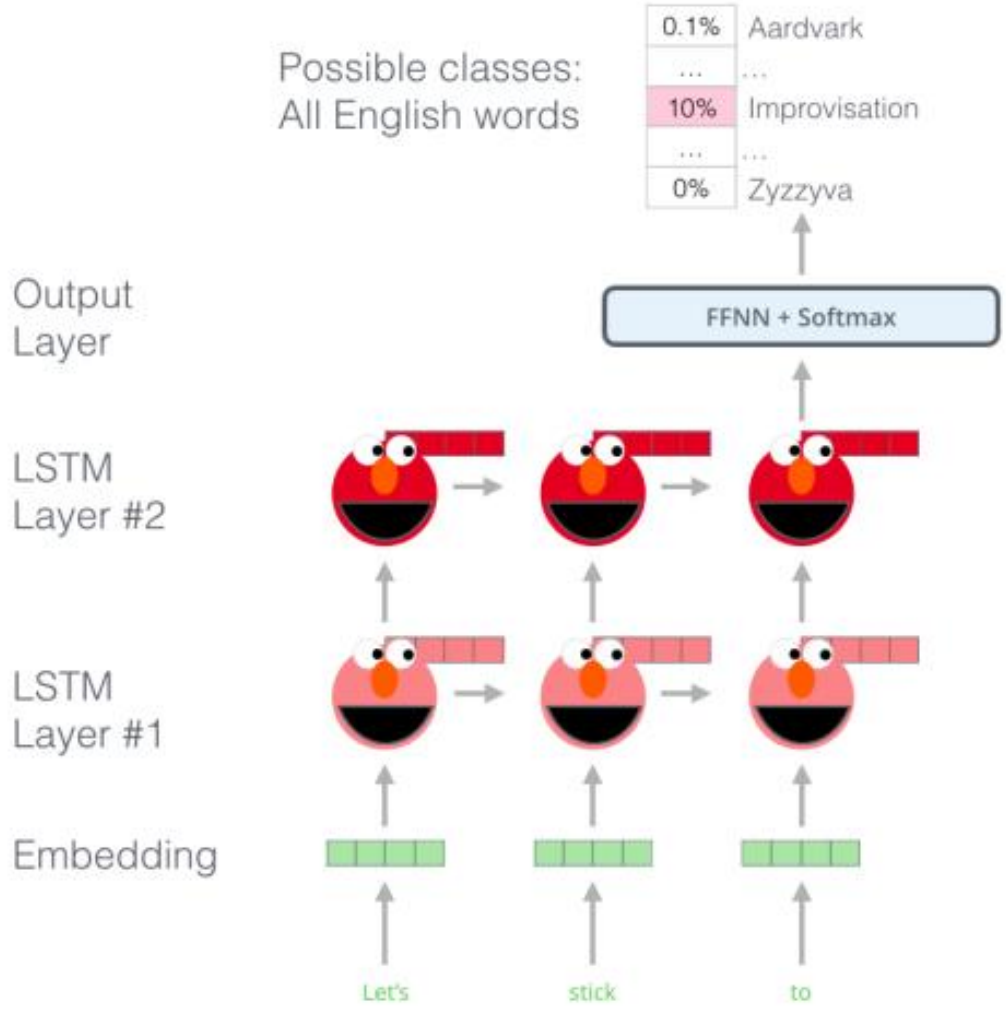
情景文本嵌入

让我们为每个单词在它的语境中编成向量

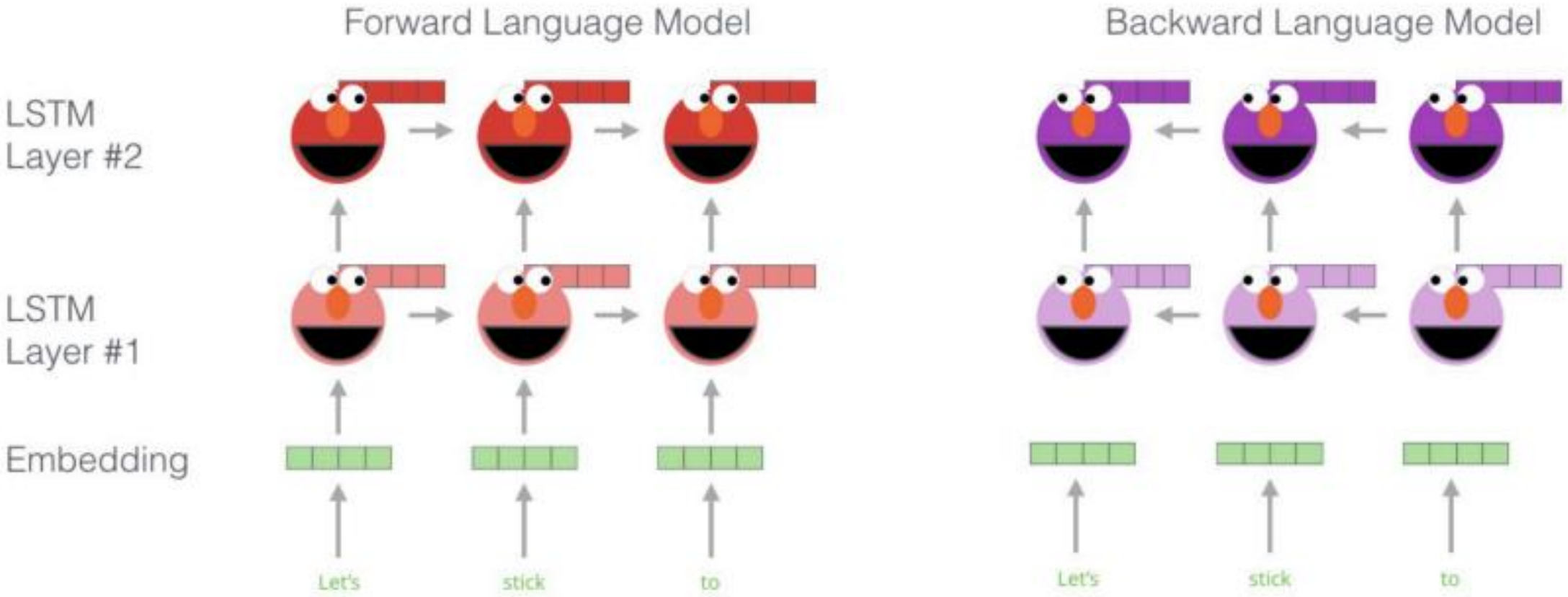
ELMo:Embeddings from Language Models

The key idea of ELMo:

在一个大型语料库上训练两个堆叠的基于LSTM的模型

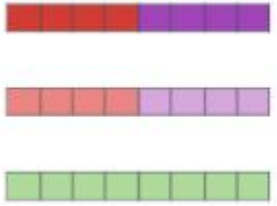


ELMo:Embeddings from Language Models



ELMo:Embeddings from Language Models

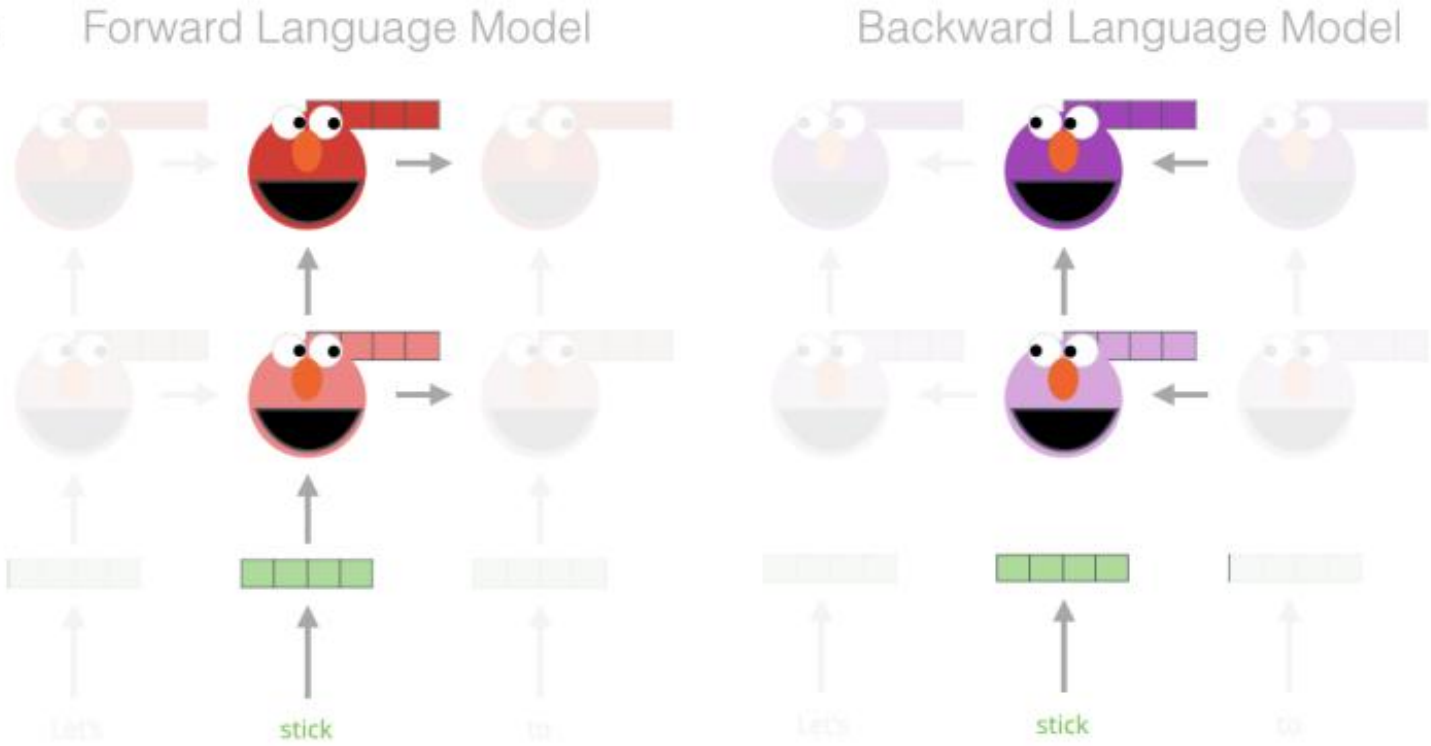
1- Concatenate hidden layers



2- Multiply each vector by a weight based on the task

3- Sum the (now weighted) vectors

ELMo embedding of "stick" for this task in this context



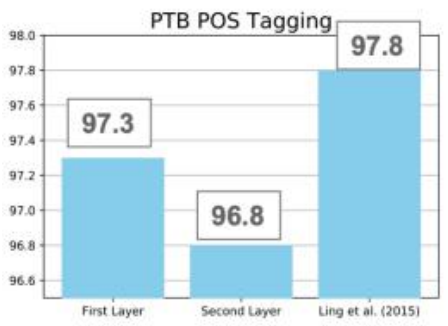
ELMo:Embeddings from Language Models

Q：为什么同时使用前向和后向语言模型呢?

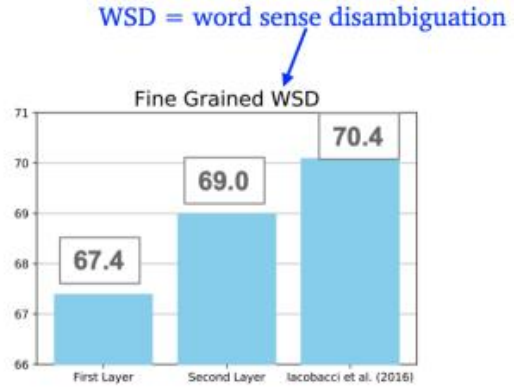
A：因为前向和后向语言模型对语境建模很重要；双向性在语言理解任务中非常重要

Q：为什么要用不同层的加权平均值呢，而不是只用顶层值?

A：因为不同层编码不同的信息



first layer > second layer

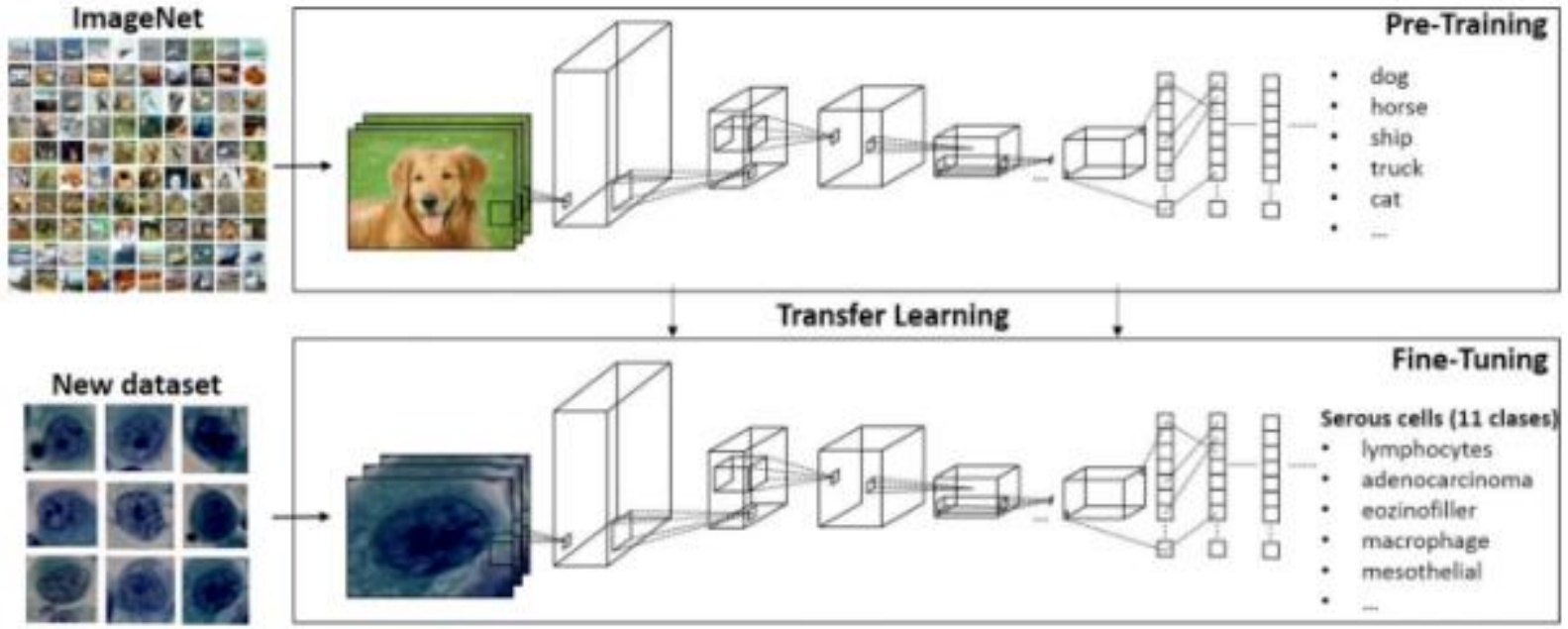


second layer > first layer

预训练

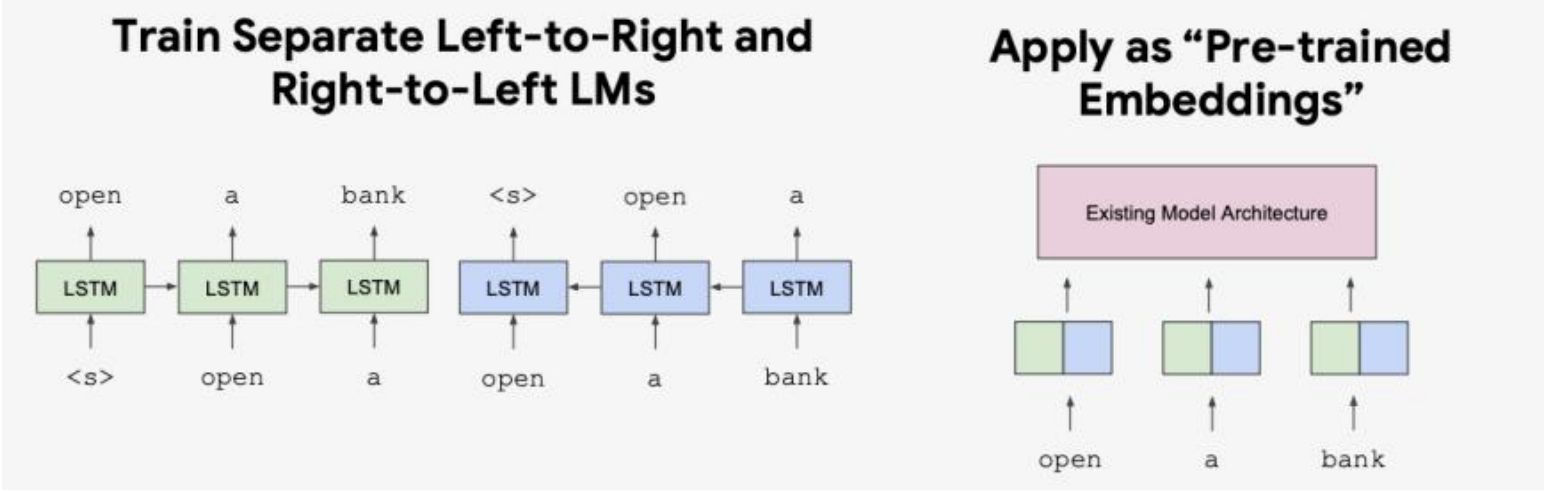
- 在任务X的大数据集上“预训练”一个模型，然后在任务Y数据集上“微调”它
- Key idea:X与Y有一些联系，所以一个能做X的模型也会对Y有一些很好的特征表示
- ImangeNet 预训练在计算机直觉领域作用巨大。

There are two existing strategies for applying pre-trained language representations to downstream tasks: feature-based and fine-tuning.

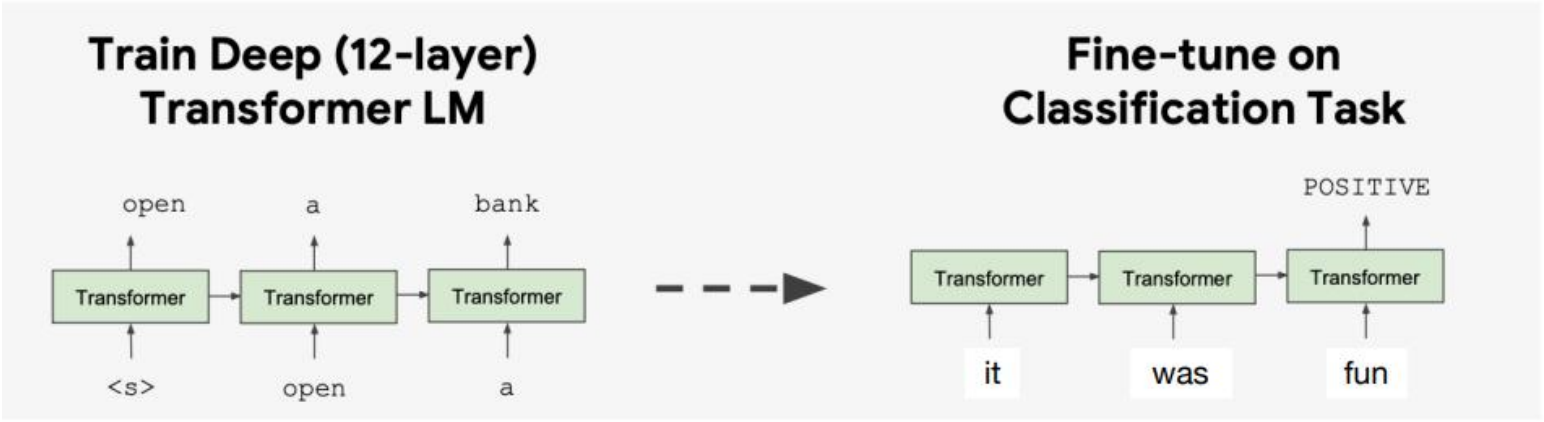


基于特征方法与微调方法

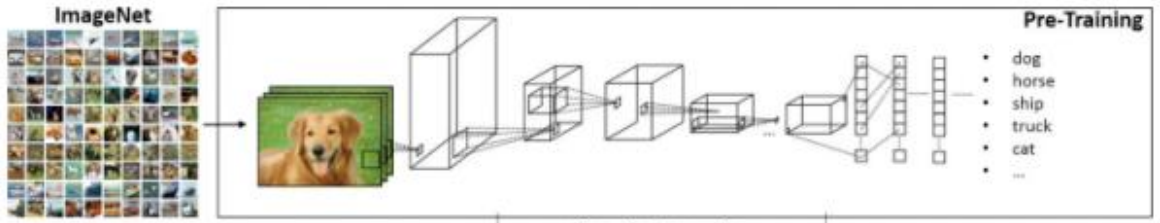
ELMo是一种基于特征的方法，它只产生可以用作现有神经网络的输入表示的词嵌入



GPT/BERT是基于微调的方法，几乎左右的模型权重都将被重用，只有少量特定与任务的权重被添加到下游任务中



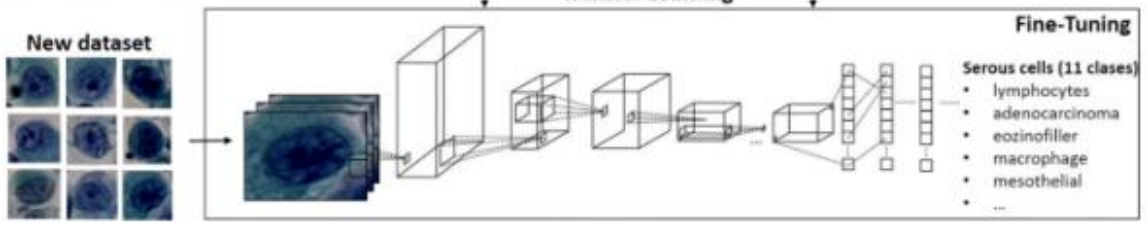
Pre-training



1.28M images, 1000 classes

Transfer Learning

Fine-tuning



3652 images, 11 classes

Pre-training

Natural language [MASK] (NLP) is an [MASK] subfield of linguistics, computer science, and artificial [MASK] concerned with the interactions [MASK] computers and human [MASK] …

→ processing, interdisciplinary, Intelligence, between, language

3.3B tokens
(512 tokens per segment)

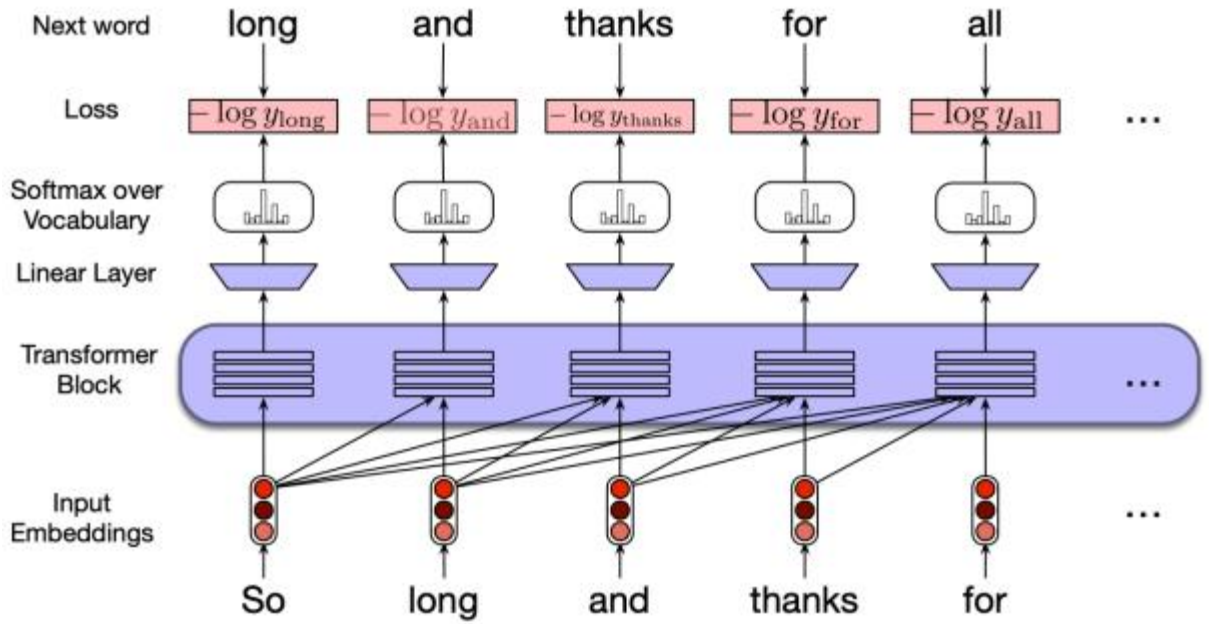
Fine-tuning

contains no wit , only labored gags	→	negative
the greatest musicians	→	positive
very good viewing alternative	→	positive

67k examples, 2 classes

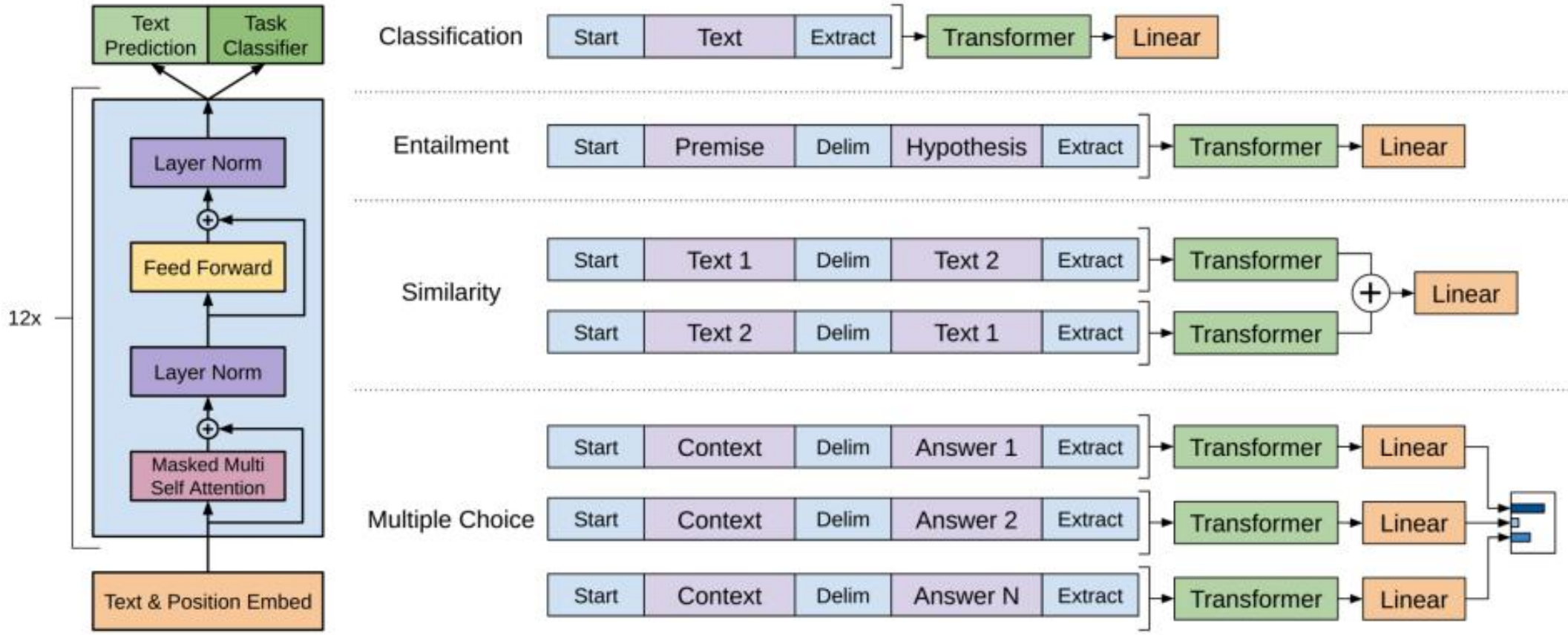
Generative Pre-Training(GPT)

- 使用Transformer decoder(unidirectional;left-to-right) instead of LSTMs
- 使用语言建模作为预训练对象
- 训练更长的文本片段（512BPE tokens）,不仅仅是单个句子



Generative Pre-Training(GPT)

- 对各种下游任务的整套模型参数进行“微调”



Transfer Learning from Supervised Data

There has also been work showing effective transfer from supervised tasks with large datasets, such as natural language inference (Conneau et al.,2017) and machine translation (McCann et al.,2017). Computer vision research has also demonstrated the importance of transfer learning fromlarge pre-trained models, where an effective recipe is to fine-tune models pre-trained with ImageNet (Deng et al., 2009; Yosinski et al., 2014).

有趣的是：……

Thanks

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

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language

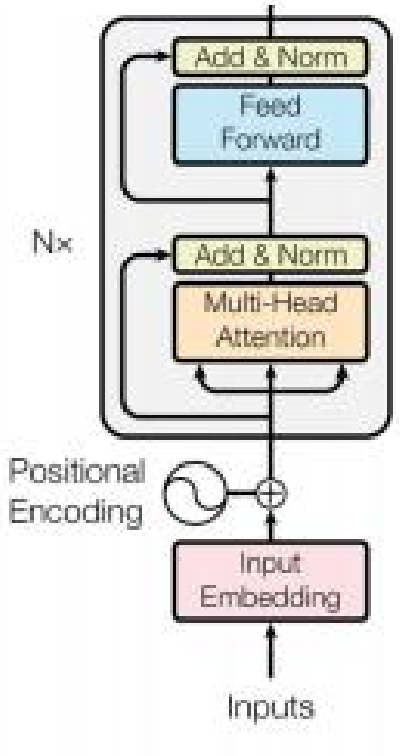
BERT: Bidirectional Encoder Representations from Transformers

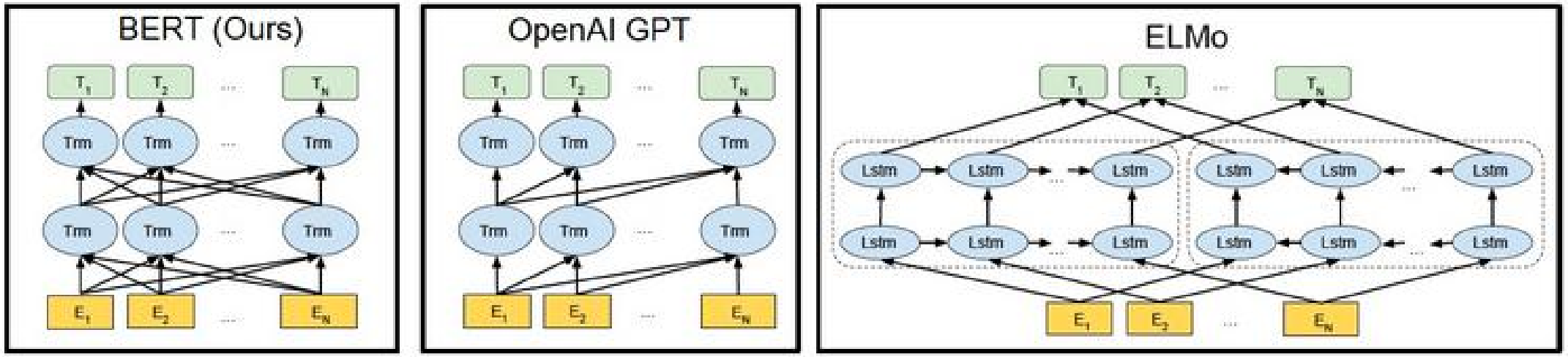
1）参考了 ELMO 模型的双向编码思想、借鉴了 GPT 用 Transformer 作为特征提取器的思路

2）GPT：GPT 使用 Transformer Decoder 作为特征提取器、具有良好的文本生成能力，然而当前词的语义只能由其前序词决定（单向的），并且在语义理解上不足

BERT：使用了 Transformer Encoder 作为特征提取器。虽然使用双向编码让 BERT 不再具有文本生成能力（机器翻译，文本摘要），但是 BERT 的语义信息提取能力更强

1）基于Transformer编码器的微调方法
2）学习基于上下文的表示
3）从大量无标记数据集中训练得到深度模型，可以显著提高各项自然语言处理任务的准确率。





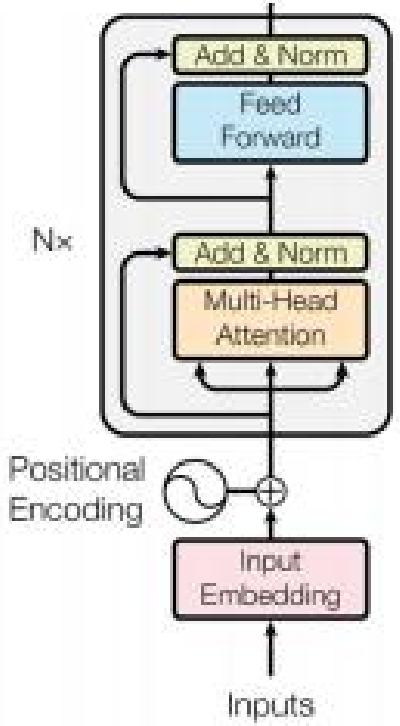
- ELMo 使用自左向右编码和自右向左编码的两个 LSTM 网络，将训练得到的特征向量以拼接的形式实现双向编码，本质上还是单向编码，只不过是两个方向上的单向编码的拼接而成的双向编码。

- GPT 使用 Transformer Decoder 用 Transformer Block 取代 LSTM 作为特征提取器，实现了单向编码，是一个标准的预训练语言模型，即使用 Fine-Tuning 模式解决下游任务。

- BERT 和 ELMo 的区别在于使用 Transformer Block 作为特征提取器，加强了语义特征提取的能力；

- BERT 和 GPT 的区别在于使用 Transformer Encoder 作为 Transformer Block，并且将 GPT 的单向编码改成双向编码，也就是说 BERT 舍弃了文本生成能力，换来了更强的语义理解能力。

两个预训练任务
1）MLM 掩码语言建模Masked language modeling
2)NSP　下一句预测Next sentence prediction

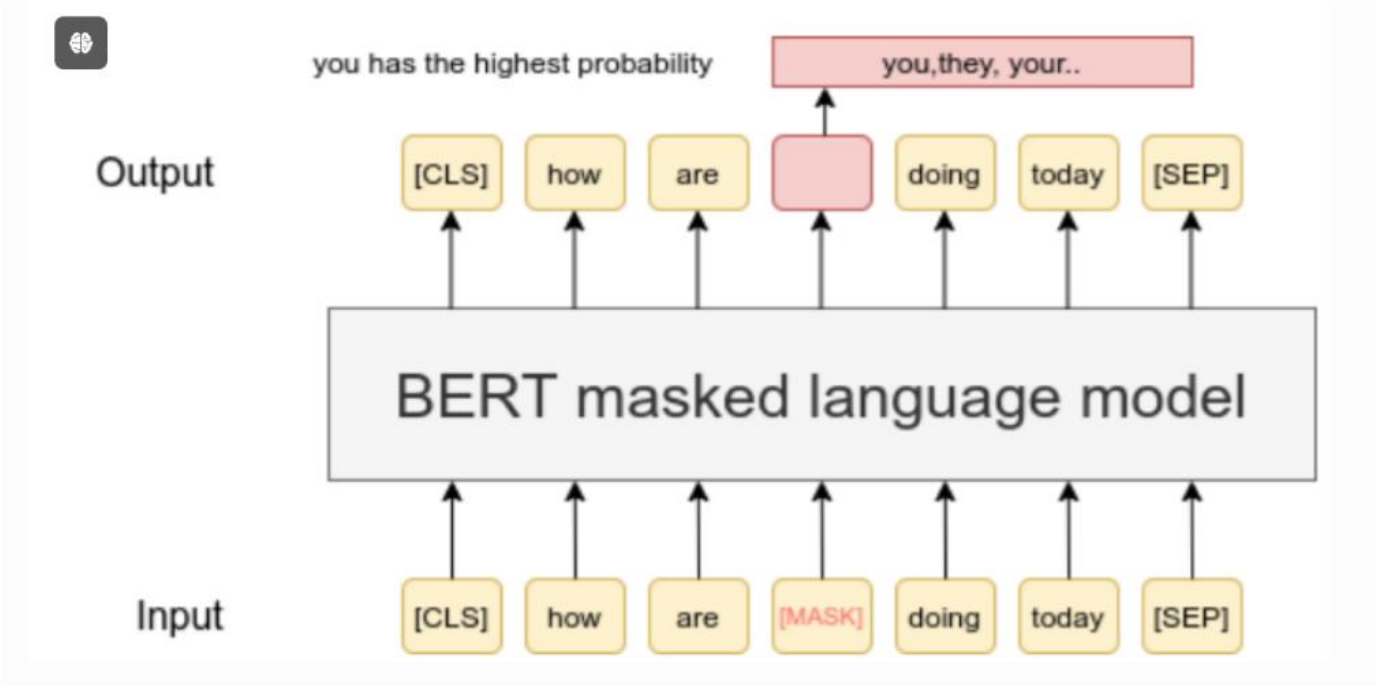


语言掩码模型（MLM）

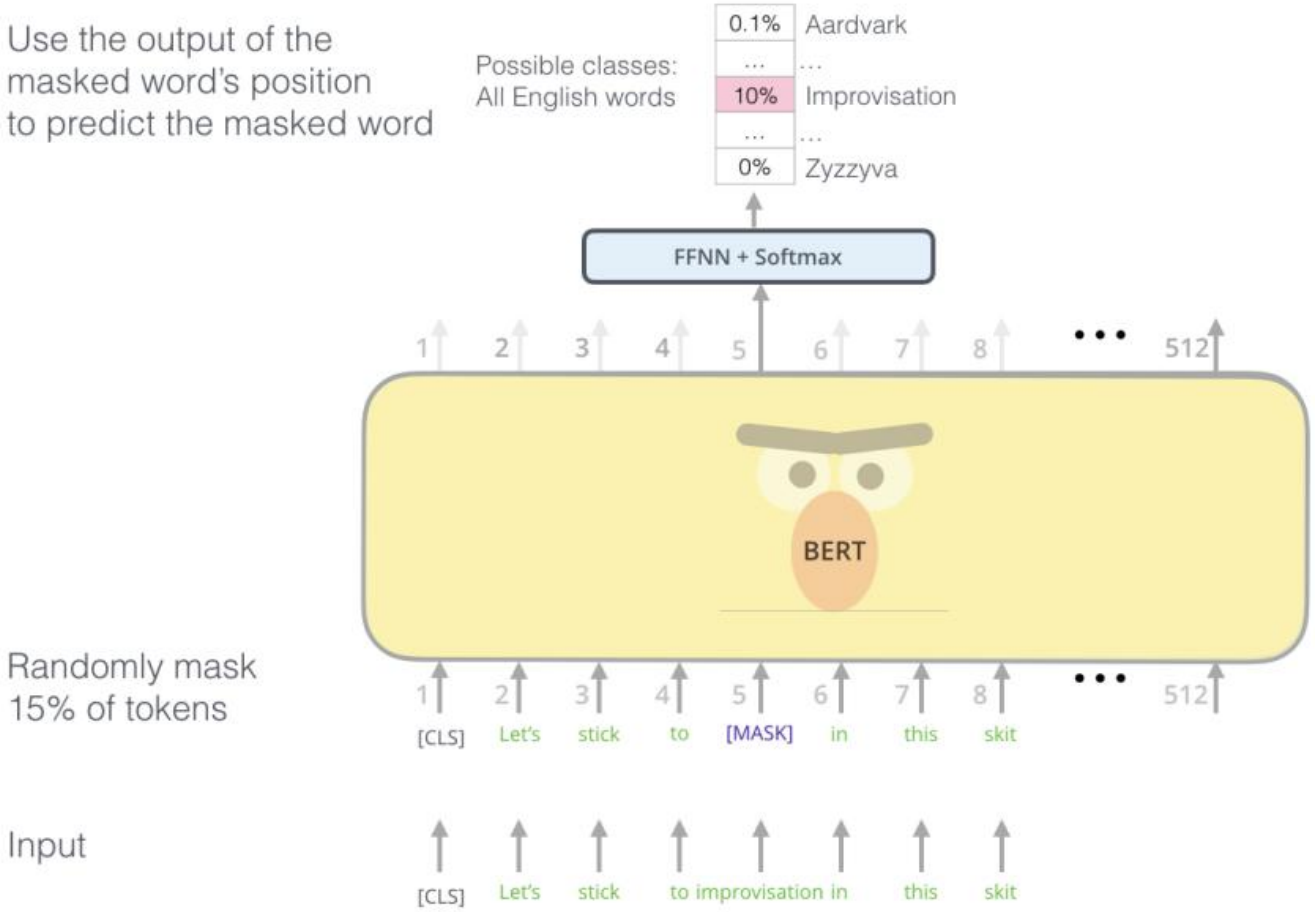
- BERT 借鉴完形填空任务的思想，使用语言掩码模型（MLM）方法训练模型。

- 随机去掉句子中的部分 token（单词），然后模型来预测被去掉的 token 是什么。

- 根据这个时刻的 hidden state 来预测这个时刻的 token 应该是什么，而不是预测下一个时刻的词的概率分布了。



- 在训练中，随机 mask 15% 的 token，这个操作则称为掩码操作。



- 弊端：由于[MASK]并不会出现在下游任务的微调（fine-tuning）阶段
- 因此预训练阶段和微调阶段之间可能产生不匹配
- 就是预训练的语言表征对[MASK]敏感，但是却对其他token不敏感。
- 因此BERT采用了以下策略来解决这个问题

无监督方法

- 考虑到上述的弊端，BERT 并没有总用 [MASK] 替换掩码词，而是按照一定比例选取替换词。
- 这是为了不让特征融合过度依赖token的标记，如果全部mask，就会过于学习token，用80%相当于引入了噪声
- 在选择 15% 的词作为掩码词后这些掩码词有三类

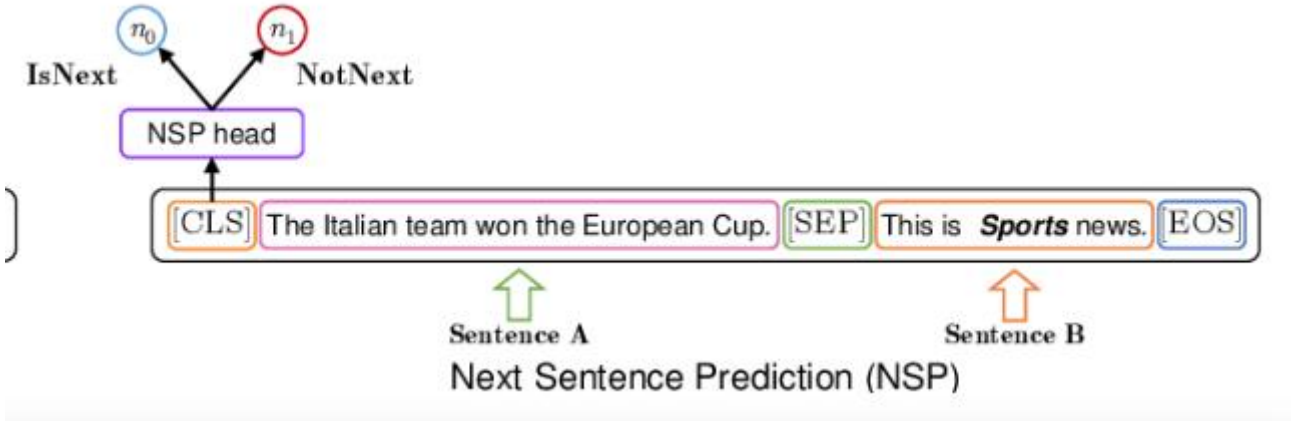
地球是太阳系八大行星之一

- 80% 练样本中：将选中的词用 [MASK] 来代替，例如：　“地球是[MASK]八大行星之一”
- 10% 的训练样本中：选中的词不发生变化，例如：“地球是太阳系八大行星之一”
- 10% 的训练样本中：将选中的词用任意的词来进行代替，：“地球是苹果八大行星之一”

- MLM是BERT能够不受单向语言模型所限制的原因。
- 简单来说就是以15%的概率用mask token （[MASK]）随机地对每一个训练序列中的token进行替换，然后预测出[MASK]位置原有的单词。

下句预测（NSP）

- 许多下游任务（问答和自然语言推断），都基于两个句子做逻辑推理，而语言模型并不具备直接捕获句子之间的语义联系的能力，
- 为了学会捕捉句子之间的语义联系，BERT 采用了下句预测（NSP）作为无监督预训练的一部分。用于减少预训练模型和微调模型的差距
- BERT 输入的语句将由两个句子构成，其中，50% 的概率将语义连贯的两个连续句子作为训练文本（连续句对一般选自篇章级别的语料，以此确保前后语句的语义强相关），另外 50% 的概率将完全随机抽取两个句子作为训练文本。

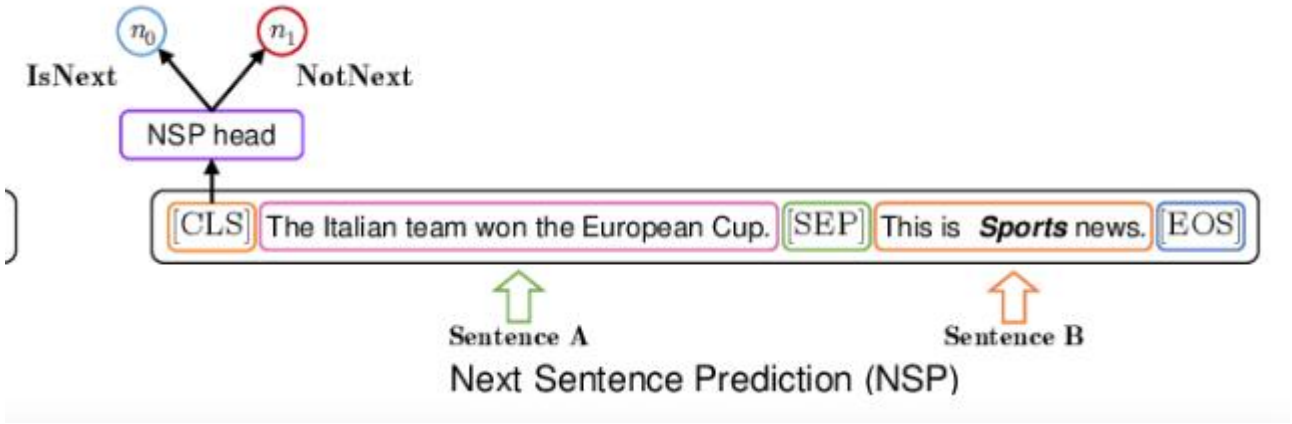


- 具体的做法是：对于每一个训练样例，在数据集中挑选出句子A和句子B来组成，50%的时候句子B就是句子A的下一句（标注为IsNext），剩下50%的时候句子B是语料库中的随机句子（标注为NotNext）。

- 接下来把训练样例输入到BERT模型中，用[CLS]对应的信息去进行二分类的预测。

连续句对：[CLS]今天天气很糟糕[SEP]下午的体育课取消了[SEP]

随机句对：[CLS]今天天气很糟糕[SEP]鱼快被烤焦啦[SEP]



- [SEP] 标签表示分隔符。 [CLS] 表示标签用于类别预测，结果为 1，表示输入为连续句对；结果为 0，表示输入为随机句对。

- 通过训练 [CLS] 编码后的输出标签，BERT 可以学会捕捉两个输入句对的文本语义，

33

预训练

Vocabulary size: 30,000 wordpieces (common sub-word units) (Wu et al., 2016)

注意力之后的 FFN 是一个 512>2048->512 的 MLP，
也就是 $8H^2$ 个参数

$8H^2$

4H—H

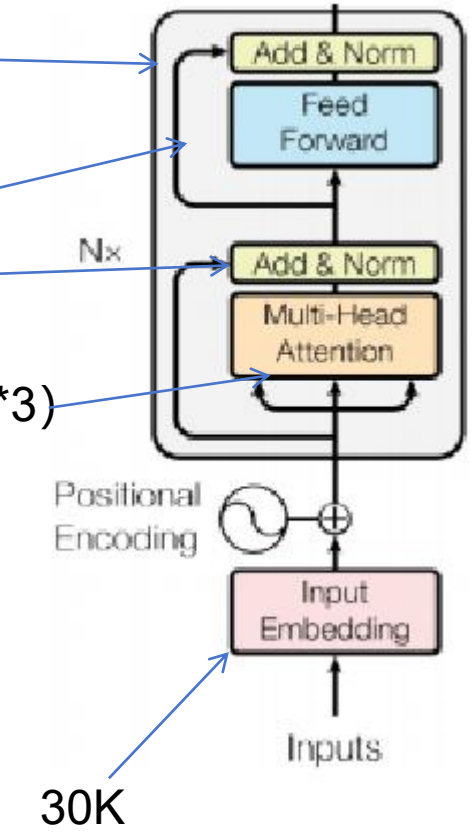
H—4H

30K*H+L*H*H*12
≈30522* 768+12* 768* 768*12)
=108808704.0
≈110M

$4H^2$

H*H

QKV（H*H*3）



- L 代表 Transformer Block 的层数；
- H 代表特征向量的维数
- A 表示 Self-Attention 的头数，
- 使用这三个参数基本可以定义 BERT的量级。
- A*64=H，

更大更强

- BERT 的模型结构其实就是 Transformer Encoder 模块的堆叠。
- 在模型参数选择上，论文给出了两套大小不一致的模型。

BERTbase：L = 12，H = 768，A = 12，
总参数量为 1.1 亿，和OpenAI的GPT一样

BERTLarge：L = 24，H = 1024，A = 16，
总参数量为 3.4 亿

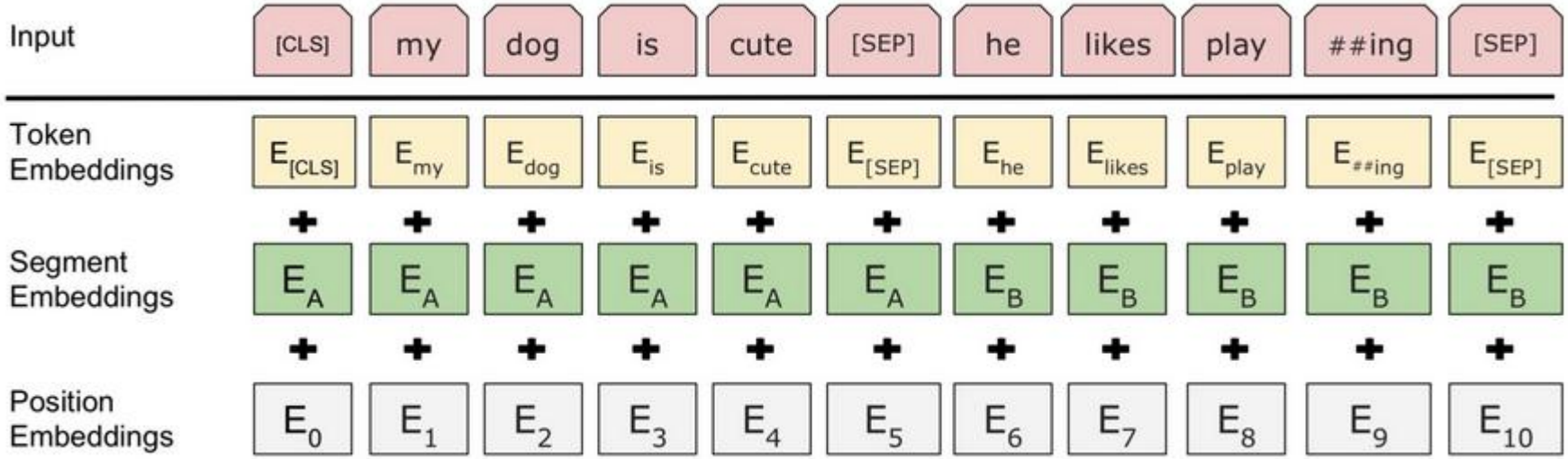
- L 代表 Transformer Block 的层数；
- H 代表特征向量的维数
- A 表示 Self-Attention 的头数，
- 使用这三个参数基本可以定义 BERT的量级。
- A*64=H，

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Average
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$BERT_{BASE}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$BERT_{LARGE}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

输入表示

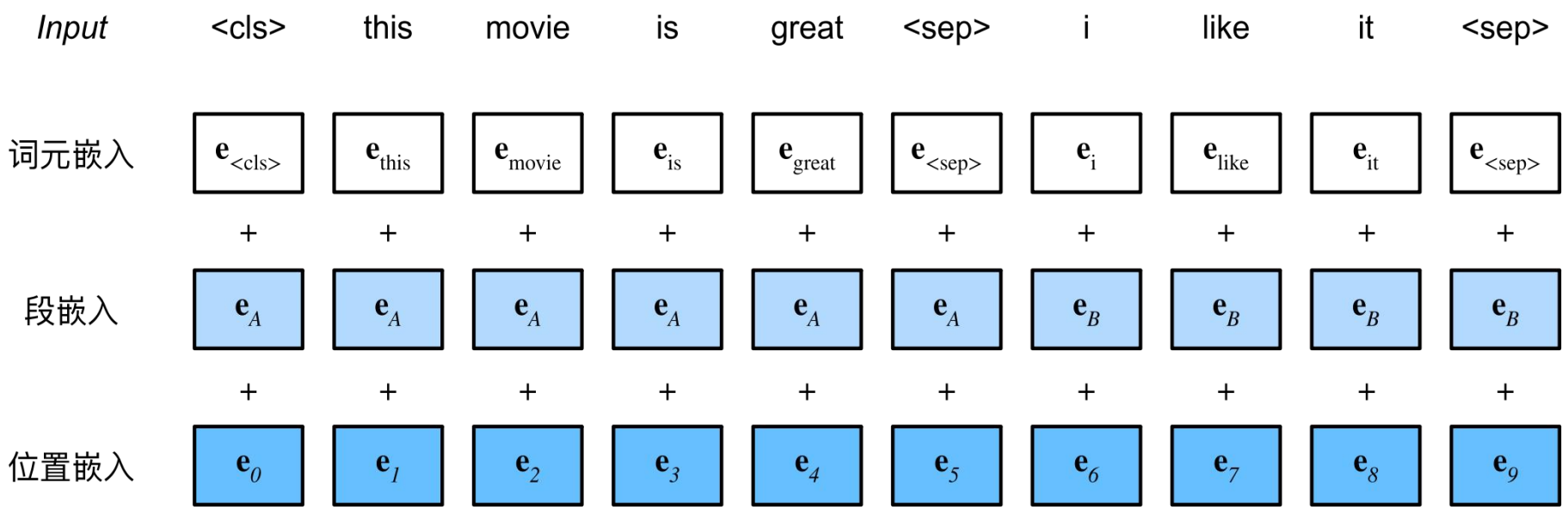
- 由于 BERT 通过 Transformer 模型堆叠而成，所以 BERT 的输入需要多套 Embedding 操作：

- 一套为 One-hot 词表映射编码（对应下图的 Token Embeddings）；
- 另一套为位置编码（对应下图的 Position Embeddings）
- 由于在 MLM 的训练过程中，存在单句输入和双句输入的情况，因此 BERT 还需要一套区分输入语句的分割编码（对应下图的 Segment Embeddings），BERT 的分割编码也将在预训练过程中训练得到

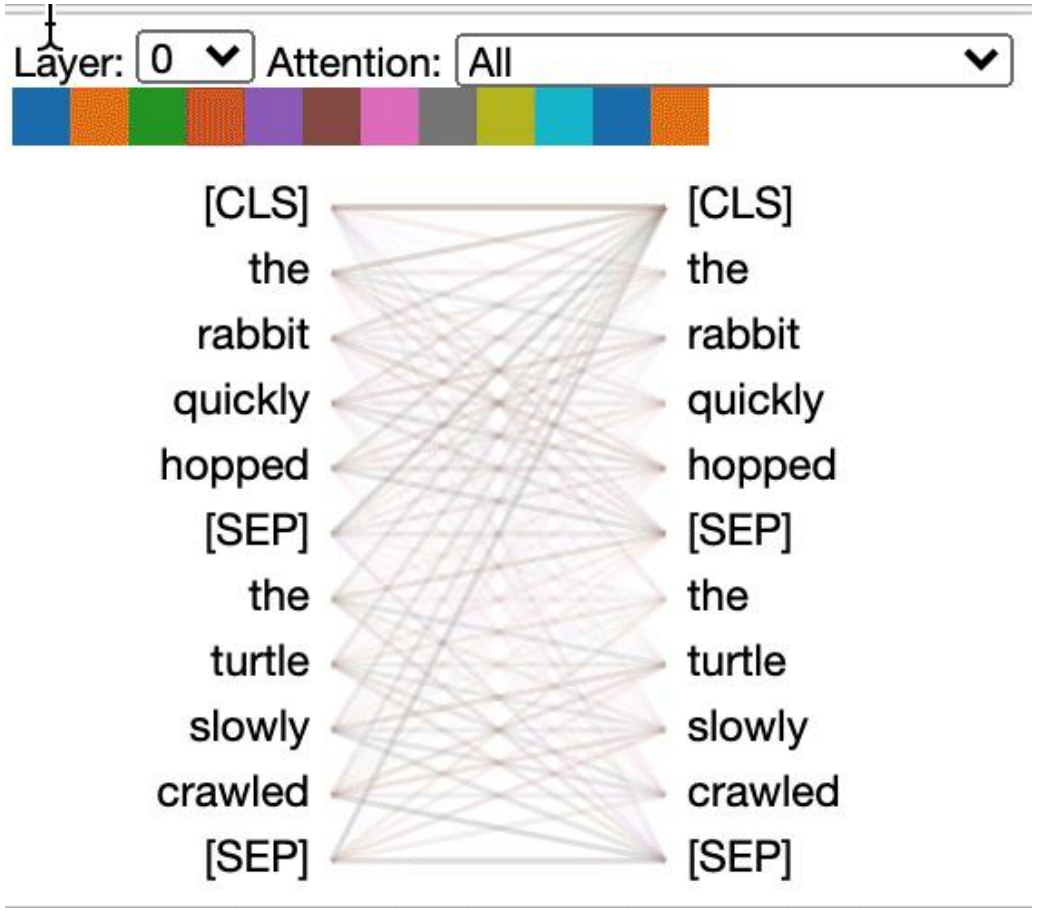


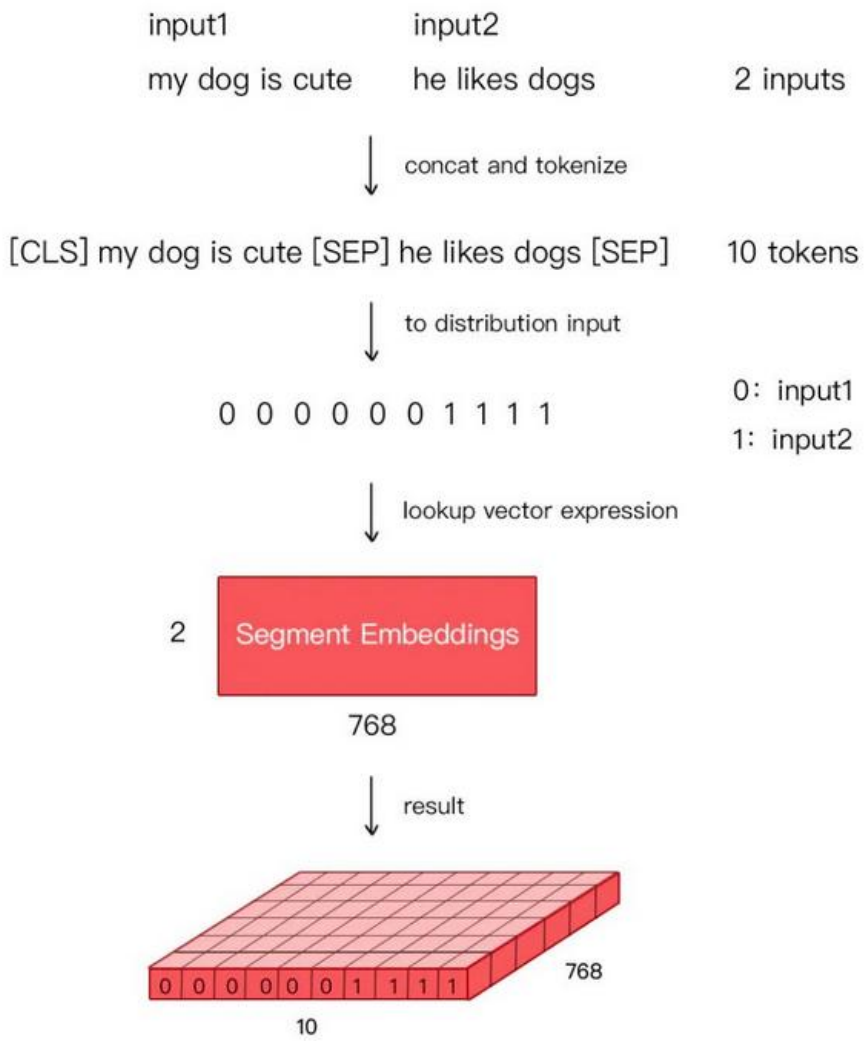
BERT是一个预训练模型，其必须要适应各种各样的自然语言任务，因此模型所输入的序列必须有能力包含一句话（文本情感分类，序列标注任务）或者两句话以上（文本摘要，自然语言推断，问答任务）。那么如何令模型有能力去分辨哪个范围是属于句子A，哪个范围是属于句子B

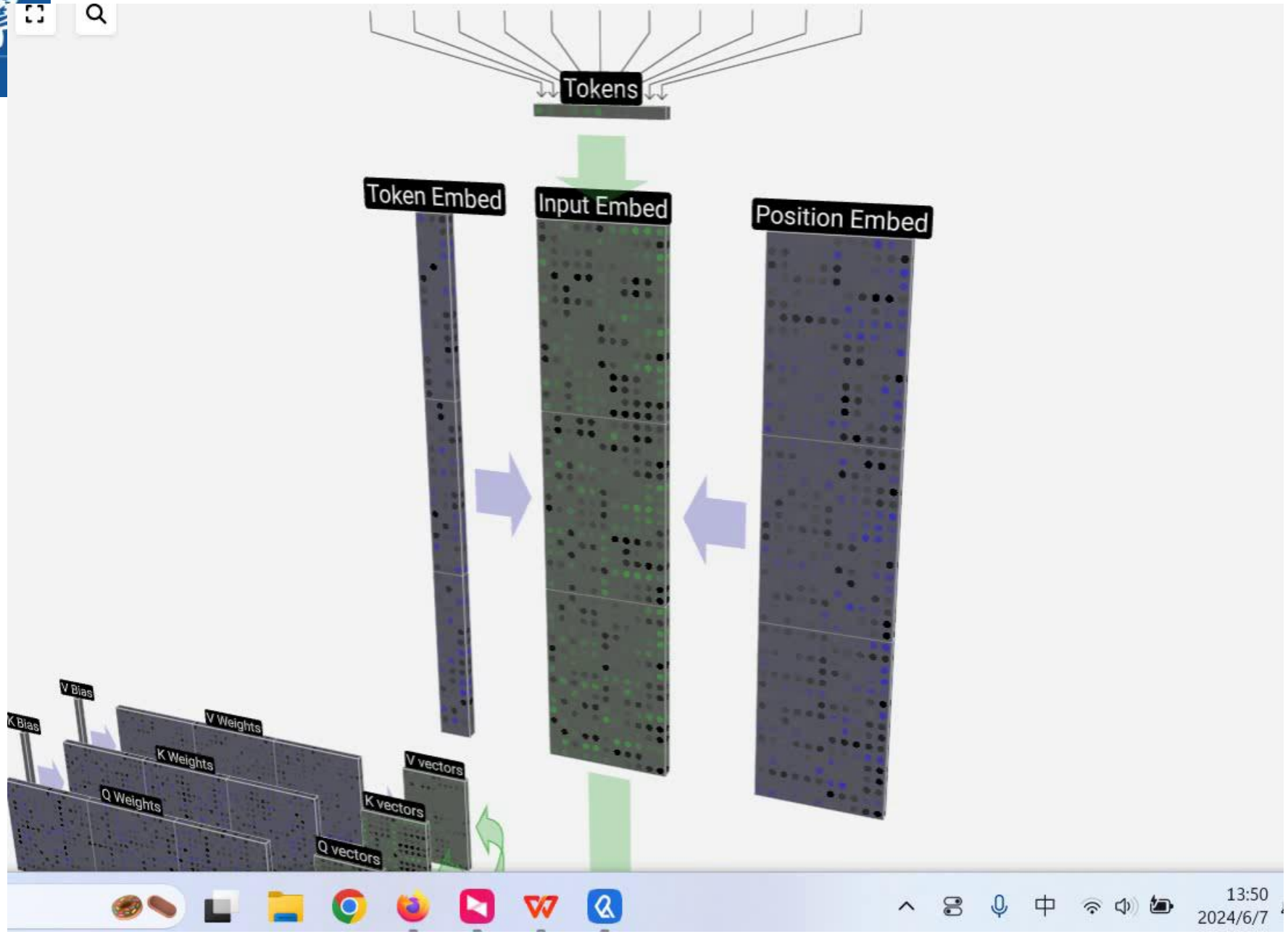
1）在序列tokens中把分割token（[SEP]）插入到每个句子后，以分开不同的句子tokens。
2）为每一个token表征都添加一个可学习的分割embedding来指示其属于句子A还是句子B。

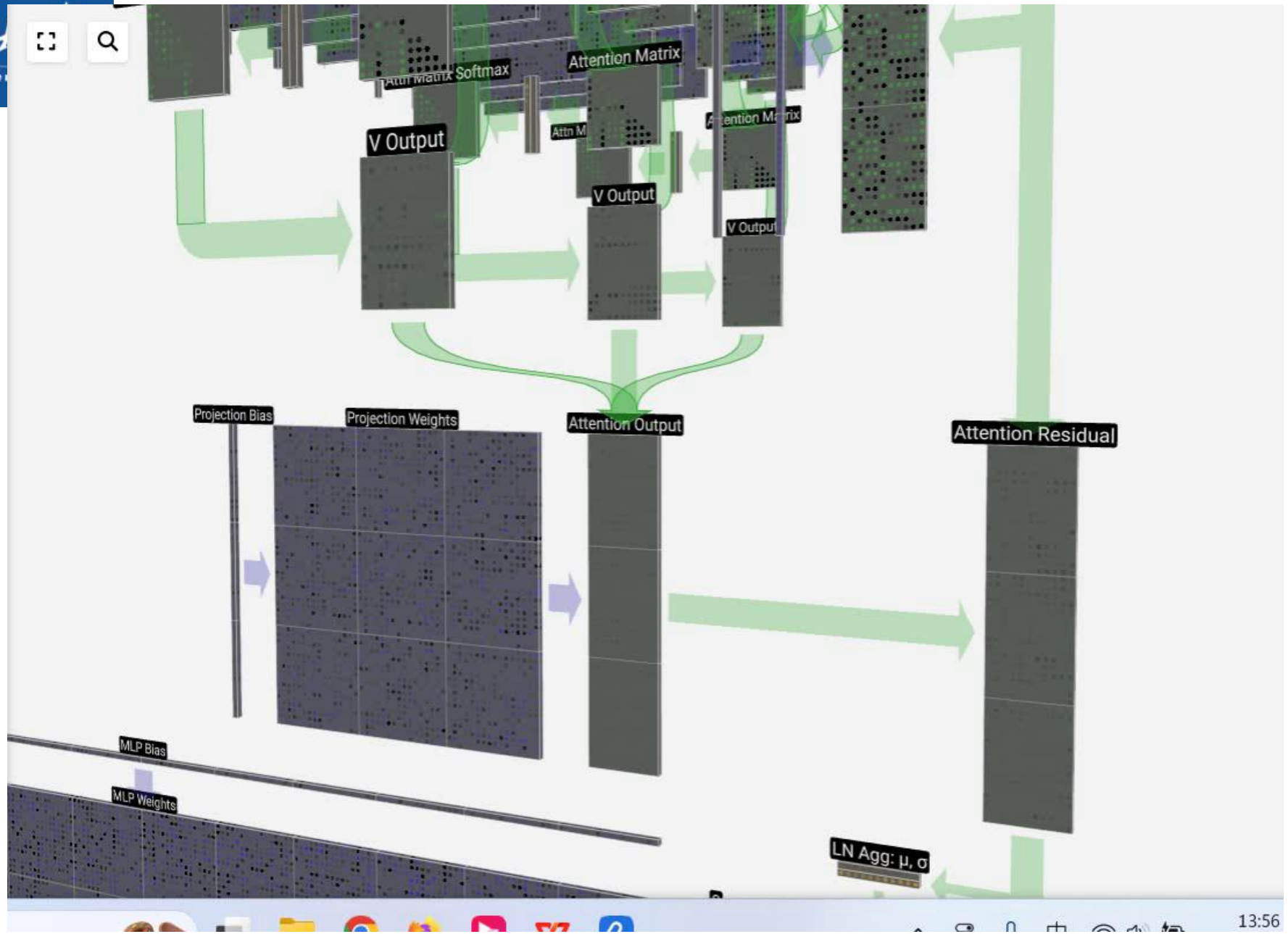


[CLS]I like dogs[SEP]I like cats[SEP]
对应编码 0 0 0 0 0 1 1 1 1
[SEP]I like dogs and cats[SEP]
对应编码 0 0 0 0 0 0 0

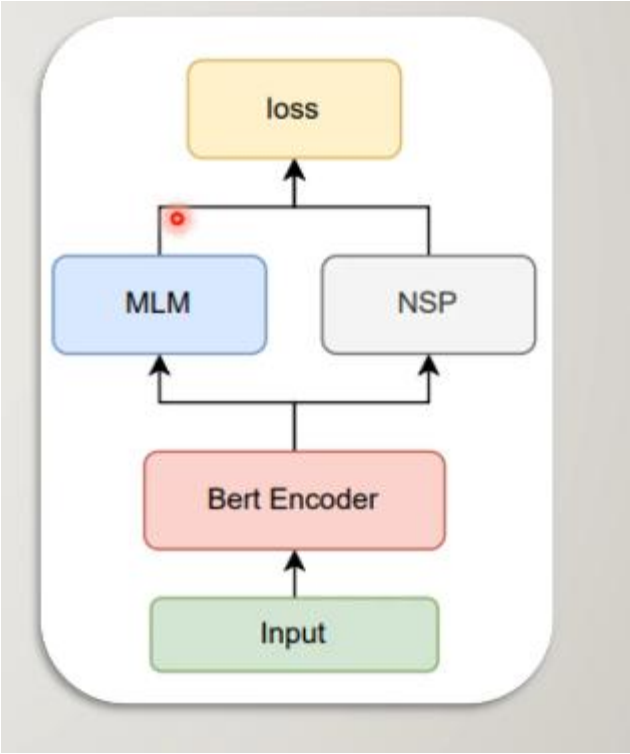




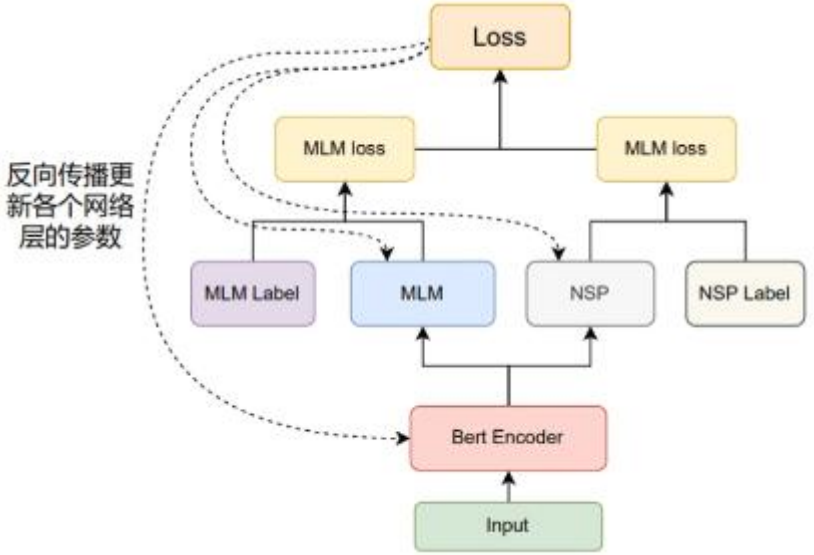

Tokens
Token Embed
Input Embed
Position Embed
V Bias
K Bias
V Weights
K Weights
Q Weights
V vectors
K vectors
Q vectors
13:50
2024/6/7


Attn Matrix Softmax
Attention Matrix
V Output
V Output
V Output
Projection Bias
Projection Weights
Attention Output
Attention Residual
MLP Bias
MLP Weights
LN Agg: μ, σ
13:56

MLM和NSP一起进行预训练



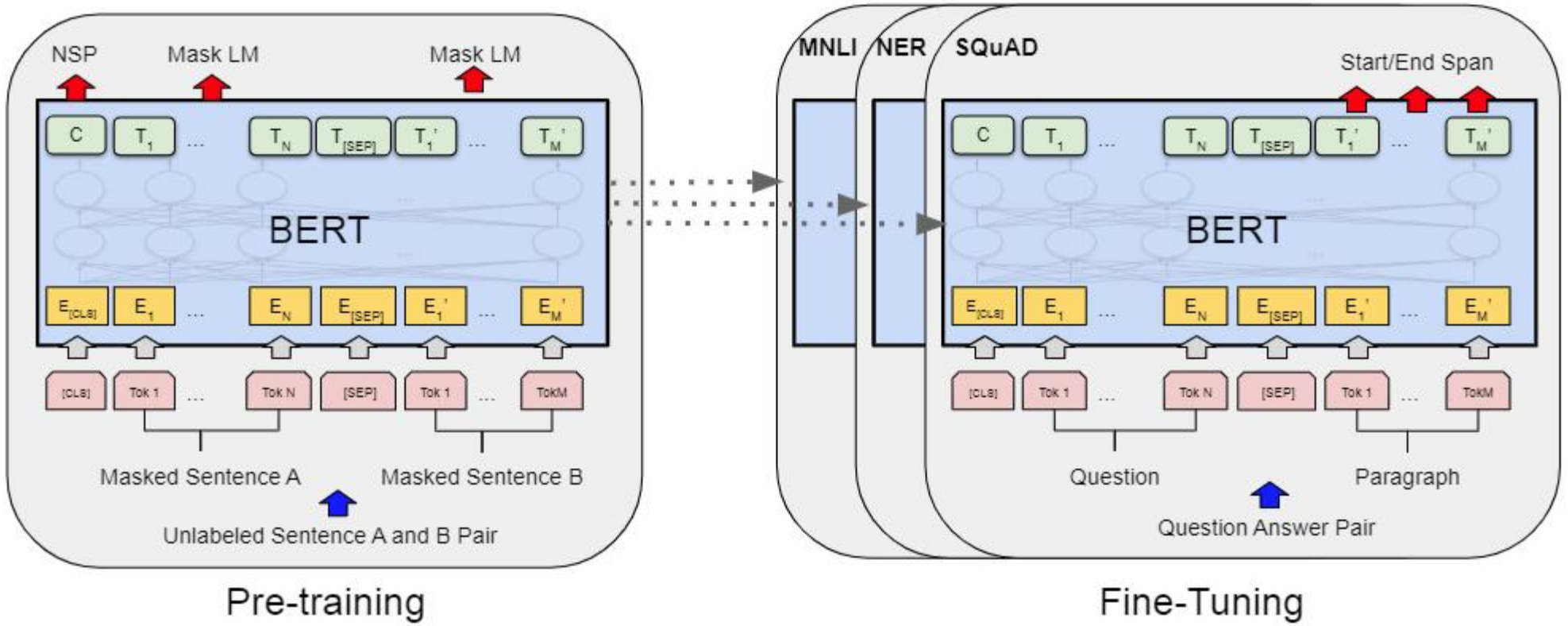
预训练一次，微调很多次



- 好处1：无监督数据无限大。不用找人标注数据，网络上所有文本都是潜在的数据集
- 好处2：能够学会语法结构、fine-tuning 能更有效率地训练下游任务
- 好处3：减少处理不同NLP任务所需的architecture engineering 成本

非监督的训练一个BERT模型，在下游任务上进行微调。CLS用于整合序列信息，SEP用于分开两个句子
和 GPT 一样，BERT 也采用二段式训练方法：

第一阶段：使用大规模无标签数据，来训练基础语言模型；
第二阶段：根据指定任务的少量带标签训练数据进行微调训练。



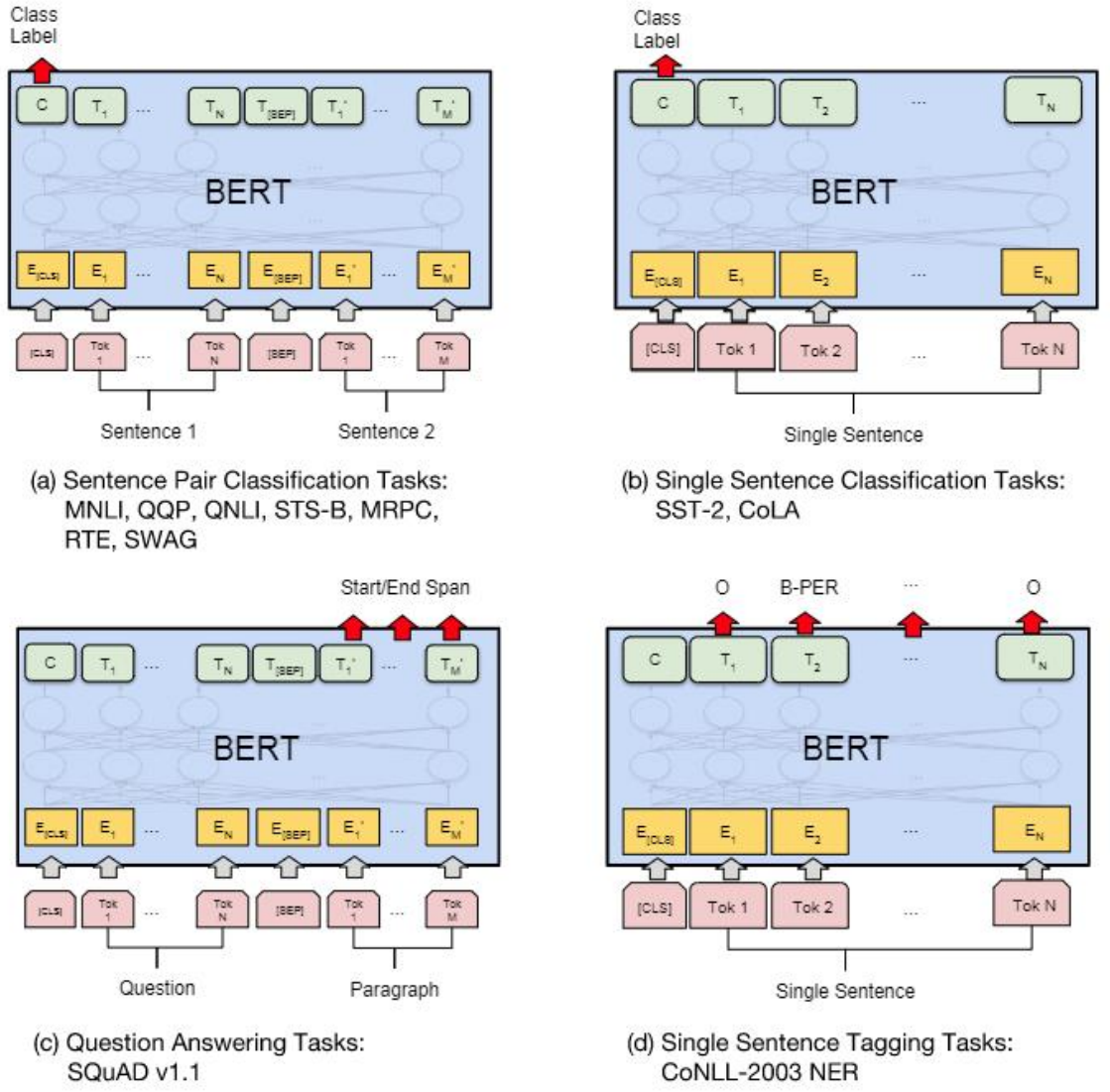
下游任务改造

十一个任务分为四大类

- 句对分类、单句分类、文本问答和单句标注

fine tune BERT 来解决新的下游任务有5 个步骤：

- 准备原始文本数据
- 将原始文本转换成BERT 相容的输入格式
- 在BERT 之上加入新layer 成下游任务模型
- 训练该下游任务模型
- 对新样本做推论



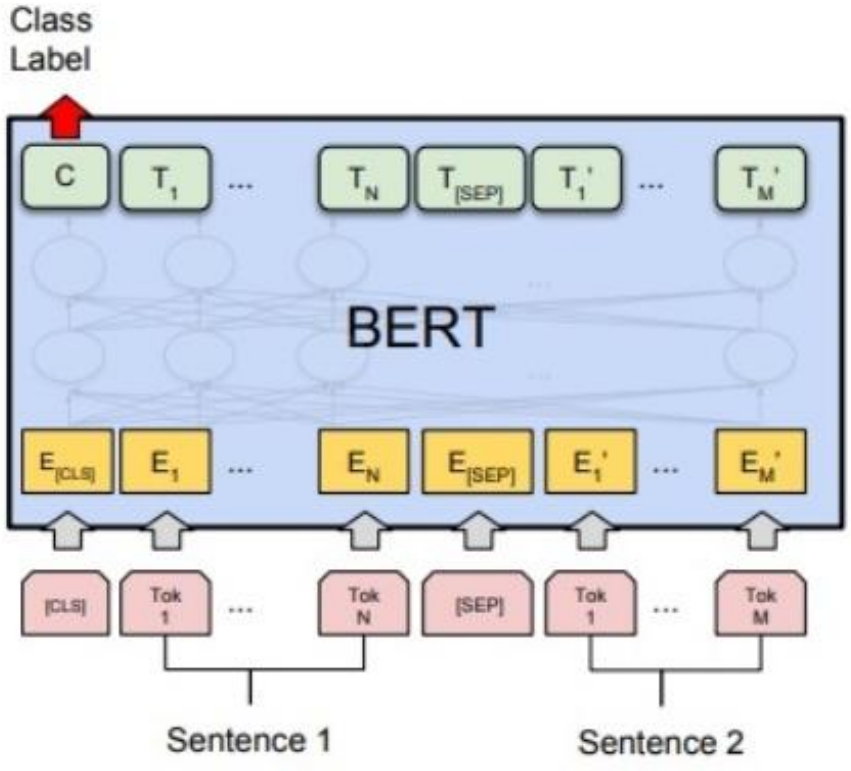
句对分类　　句子级别的任务

给定两个句子，判断它们的关系，称为句对分类，
例如判断句对是否相似、判断后者是否为前者的答案。

- 针对句对分类任务，BERT 在预训练过程中就使用了 NSP 训练方法获得了直接捕获句对语义关系的能力。

如下图所示，

- 句对用 [SEP] 分隔符拼接成文本序列，
- 在句首加入标签 [CLS]，
- 将句首标签所对应的输出值作为分类标签，
- 计算预测分类标签与真实分类标签的交叉熵，将其作为优化目标，在任务数据上进行微调训练。



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

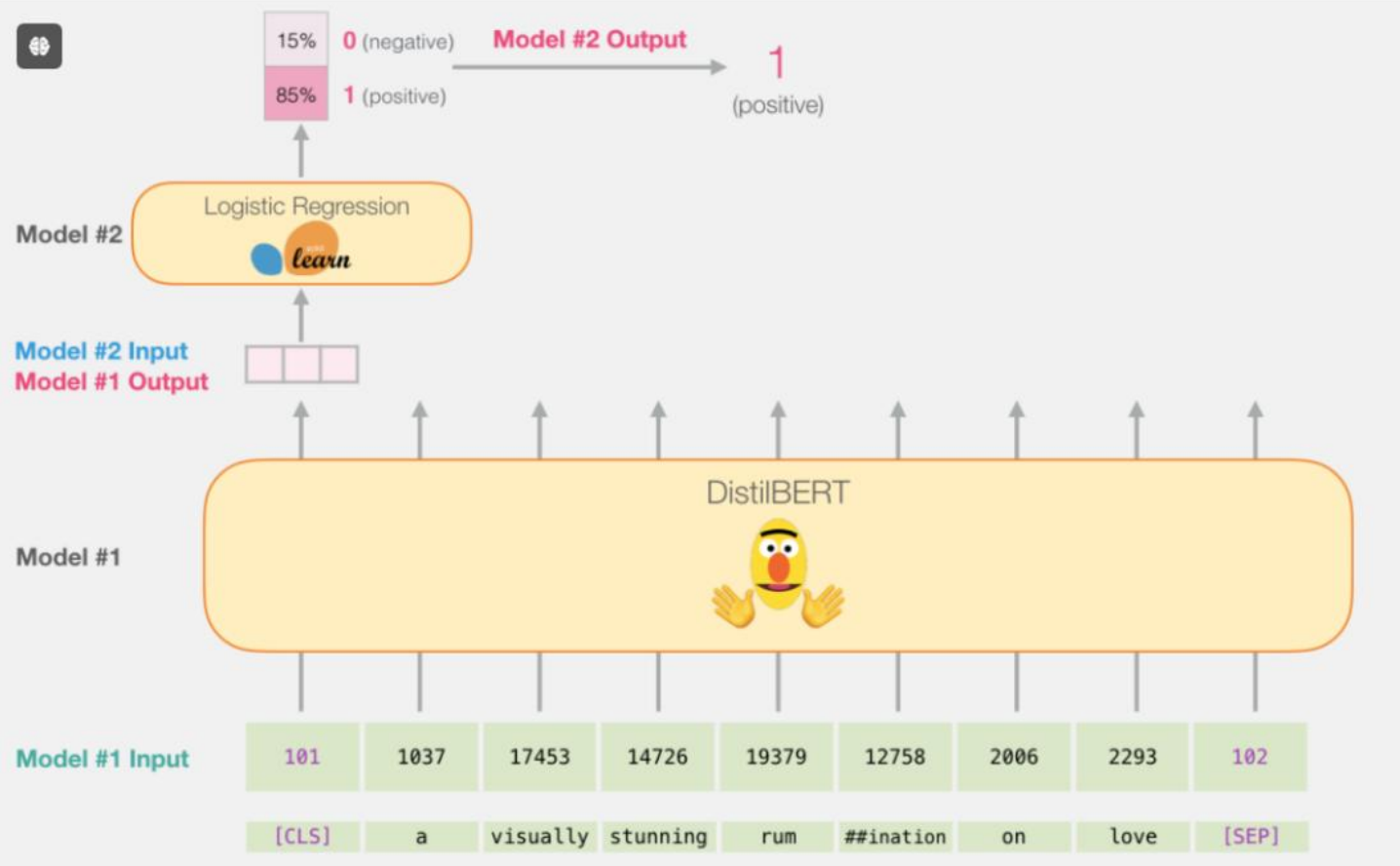
- 二分类任务，BERT 不需要对输入数据和输出数据的结构做任何改动

任务：判断句子 “今天38度” 和句子 “今天很热” 是否相似

输入改写： “[CLS]今天38度[SEP]今天很热”

取 “[CLS]” 标签对应输出：[0.02, 0.98]

通过 arg max 操作得到相似类别为 1，即两个句子相似

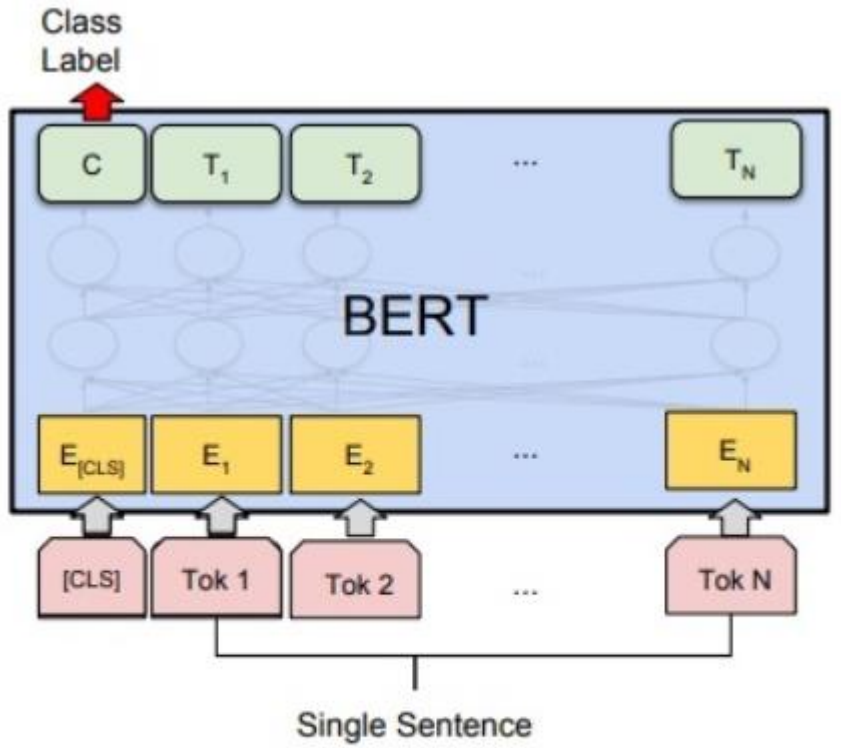


单句分类

- 给定一个句子，判断该句子的类别，统称为单句分类，例如判断情感类别、判断是否为语义连贯的句子。

下面给出语义连贯性判断任务的实例：
任务：判断句子“海大球星饭茶吃” 是否为一句话
输入改写： “[CLS]海大球星饭茶吃”
取 “[CLS]” 标签对应输出：[0.99, 0.01]

通过 arg max 操作得到相似类别为 0，即这个句子不是一个语义连贯的句子

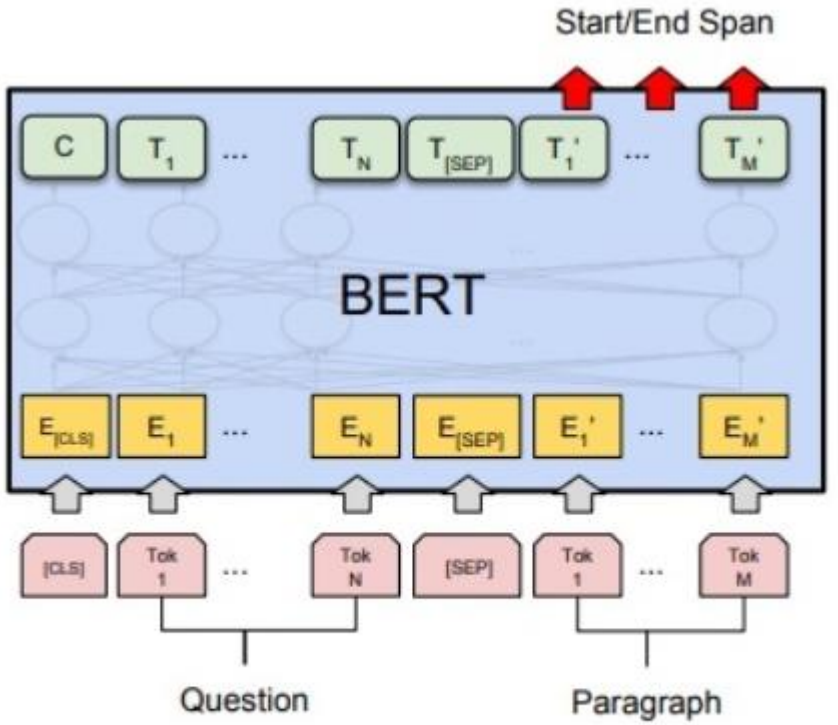


(b) Single Sentence Classification Tasks: SST-2, CoLA

文本问答 Token级别的任务

- 给定一个问句和一个蕴含答案的句子，找出答案在后这种的位置，称为文本问答，
- 例如给定一个问题（句子 A），在给定的段落（句子 B）中标注答案的其实位置和终止位置。

- 为了标注答案的起始位置和终止位置，BERT 引入两个辅助向量
- s（start，判断答案的起始位置） 和 e（end，判断答案的终止位置）。



(c) Question Answering Tasks: SQuAD v1.1

文本回答任务的微调训练使用了两个技巧：

- 用全连接层把 BERT 提取后的深层特征向量转化为用于判断答案位置的特征向量
- 引入辅助向量 s 和 e 作为答案其实位置和终止位置的基准向量，明确优化目标的方向和度量方法

下面给出文本问答任务的实例：

任务：给定问句 “今天的最高温度是多少”，在文本 “天气预报显示今天最高温度 37 摄氏度” 中标注答案的起始位置和终止位置

输入改写： “[CLS]今天的最高温度是多少[SEP]天气预报显示今天最高温度 37 摄氏度”

BERT Softmax 结果：

篇章文本	天气	预报	显示	今天	最高温	37	摄氏度
起始位置概率	0.01	0.01	0.01	0.04	0.10	0.80	0.03
终止位置概率	0.01	0.01	0.01	0.03	0.04	0.10	0.80

对 Softmax 的结果取 arg max，得到答案的起始位置为 6，
终止位置为 7，即答案为 “37 摄氏度”

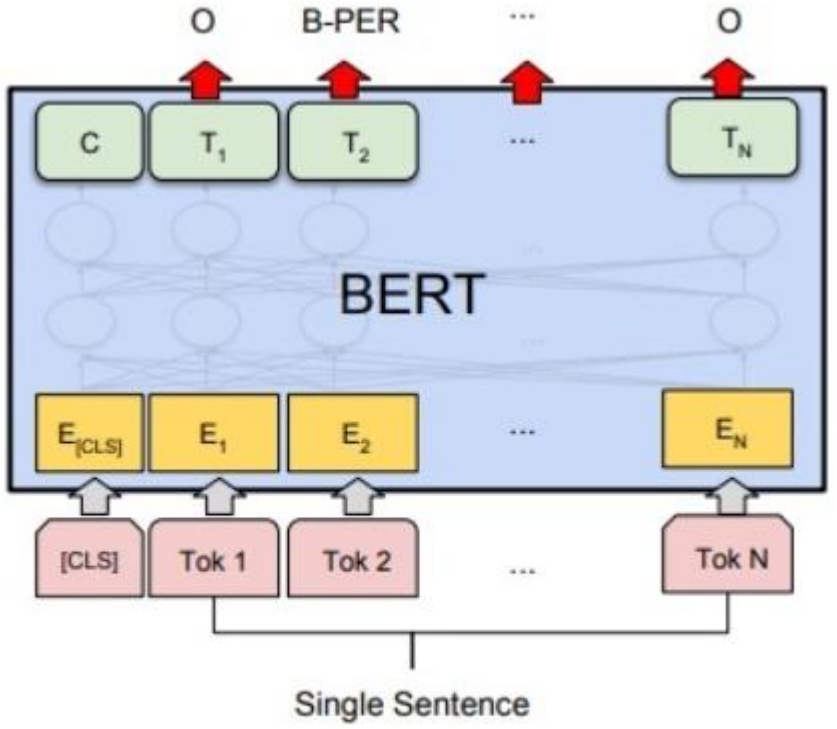
单句标注

- 给定一个句子，标注每个词的标签，称为单句标注。

例如给定一个句子，标注句子中的人名、地名和机构名。

- 单句标注任务和 BERT 预训练任务具有较大差异，但与文本问答任务较为相似。

- 在进行单句标注任务时，需要在每个词的最终语义特征向量之后添加全连接层，将语义特征转化为序列标注任务所需的特征，
- 单句标注任务需要对每个词都做标注，因此不需要引入辅助向量，直接对经过全连接层后的结果做 Softmax 操作，即可得到各类标签的概率分布。



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

由于 BERT 需要对输入文本进行分词操作，独立词将会被分成若干子词，因此 BERT 预测的结果将会是 5 类（细分为 13 小类）：

O（非人名地名机构名，O 表示 Other）
B-PER/LOC/ORG（人名/地名/机构名初始单词，B 表示 Begin）
I-PER/LOC/ORG（人名/地名/机构名中间单词，I 表示 Intermediate）
E-PER/LOC/ORG（人名/地名/机构名终止单词，E 表示 End）
S-PER/LOC/ORG（人名/地名/机构名独立单词，S 表示 Single）

将 5 大类的首字母结合，可得 IOBES，这是序列标注最常用的标注方法。

命名实体识别（NER）任务的示例：

任务：给定句子 “爱因斯坦在柏林发表演讲”，根据 IOBES 标注 NER 结果

输入改写： “[CLS]爱 因 斯坦 在 柏林 发表 演讲”

BERT Softmax 结果：

BOBES	爱	因	斯坦	在	柏林	发表	演讲
O	0.01	0.01	0.01	0.90	0.01	0.90	0.90
B-PER	0.90	0.01	0.01	0.01	0.01	0.01	0.01
I-PER	0.01	0.90	0.01	0.01	0.01	0.01	0.01
E-PER	0.01	0.01	0.90	0.01	0.01	0.01	0.01
S-LOC	0.01	0.01	0.01	0.01	0.01	0.01	0.01

对 Softmax 的结果取 argmax，得到最终地 NER 标注结果为： “爱因斯坦” 是人名；

NLP 四大类任务都可以比较方便地改造成 Bert 能够接受的方式

不同类型的任务需要对模型做不同的修改，但是修改都是非常简单的，最多加一层神经网络即可

这意味着它几乎可以做任何NLP的下游任务，具备普适性，这是很强的。

对比实验

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

更大更强

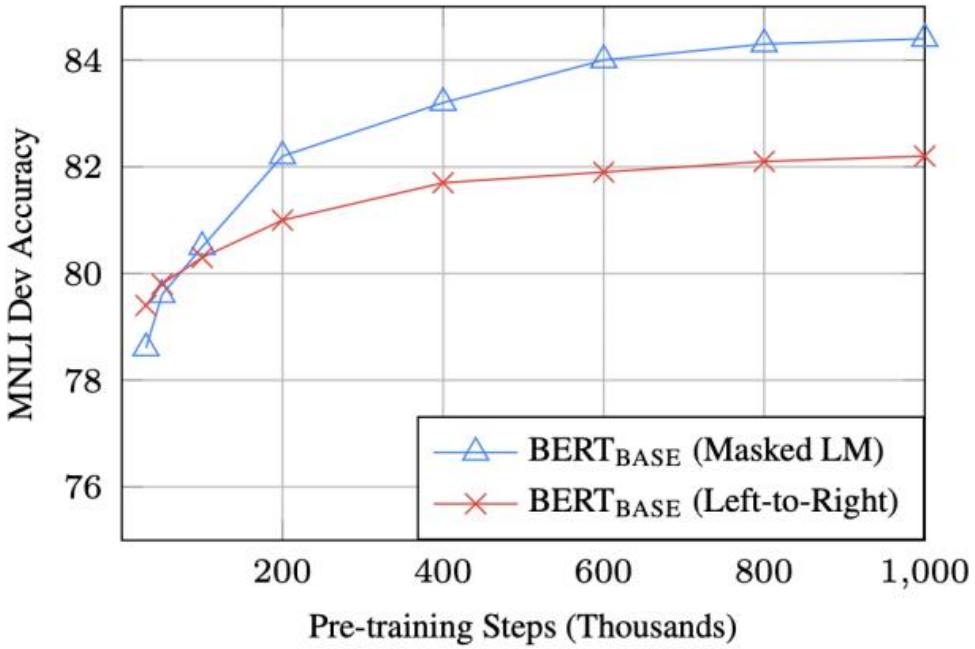
对比实验

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$BERT_{BASE}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$BERT_{LARGE}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

消融实验

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

每一个模块都很有用



MLM需要更长的收敛时间
因为有15%的mask

Thanks