GPT

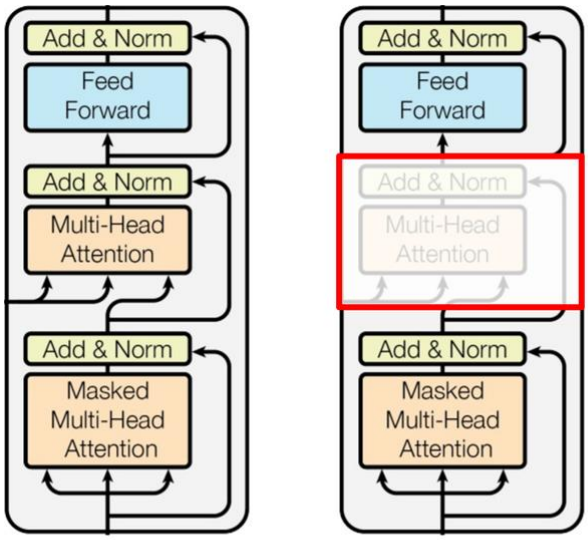
来源：《Improving Language Understanding by Generative Pre-Training》

全称：Generative Pre-Training

两个阶段：第一个阶段是利用语言模型进行预训练（无监督形式），第二阶段通过 Fine-tuning 的模式解决下游任务（监督模式下）

模型概述

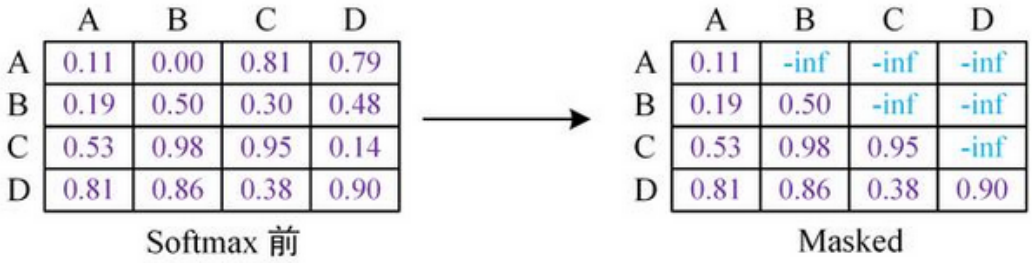
- **模型结构**



Transformer Decoder　　GPT Decoder 去掉第二个 Multi-Head

简而言之就是Transformer的Decoder去掉了中间的多头注意力，只留下了Masked的多头注意力。

  - Masked Multi-Head Attention



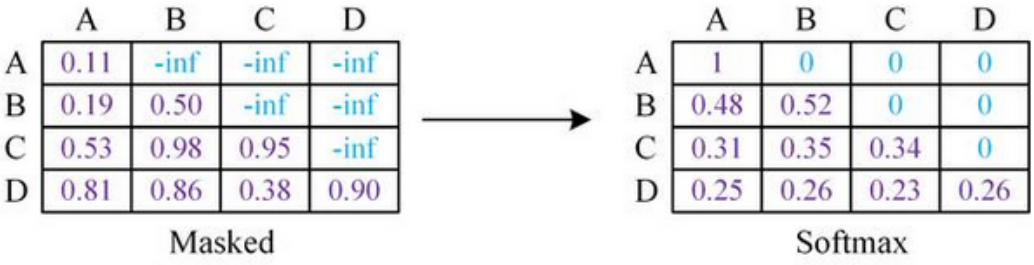
	A	B	C	D
A	0.11	0.00	0.81	0.79
B	0.19	0.50	0.30	0.48
C	0.53	0.98	0.95	0.14
D	0.81	0.86	0.38	0.90

Softmax 前

	A	B	C	D
A	0.11	-inf	-inf	-inf
B	0.19	0.50	-inf	-inf
C	0.53	0.98	0.95	-inf
D	0.81	0.86	0.38	0.90

Masked

Softmax 之前需要 Mask



	A	B	C	D
A	0.11	-inf	-inf	-inf
B	0.19	0.50	-inf	-inf
C	0.53	0.98	0.95	-inf
D	0.81	0.86	0.38	0.90

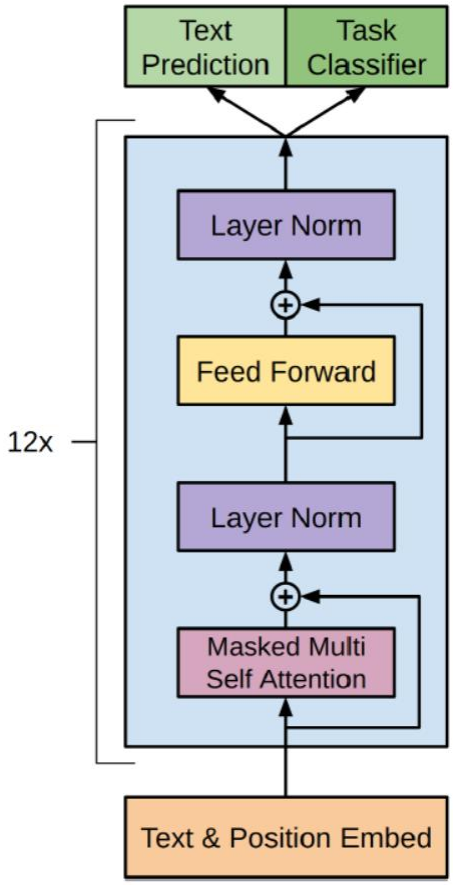
Masked

	A	B	C	D
A	1	0	0	0
B	0.48	0.52	0	0
C	0.31	0.35	0.34	0
D	0.25	0.26	0.23	0.26

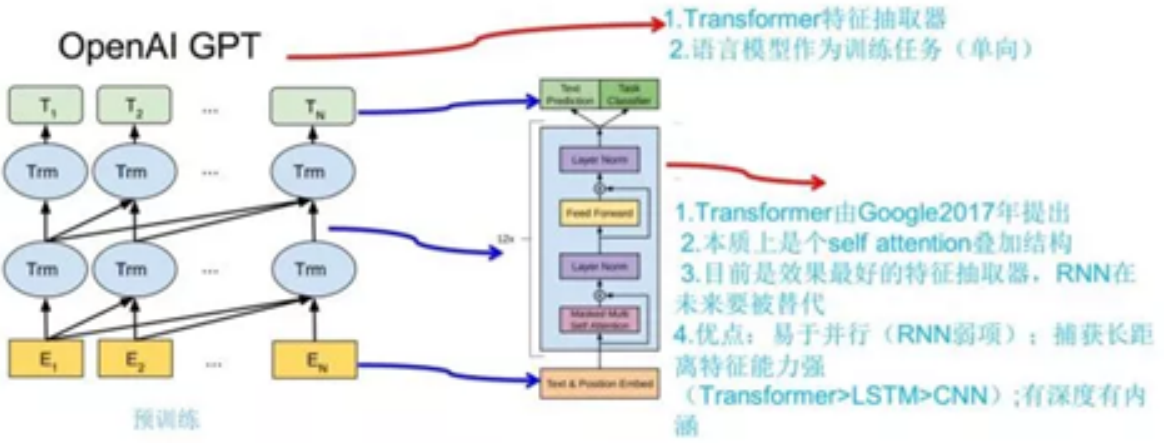
Softmax

GPT Softmax

- GPT 整体模型图，其中包含了 12 个 Decoder



预训练过程

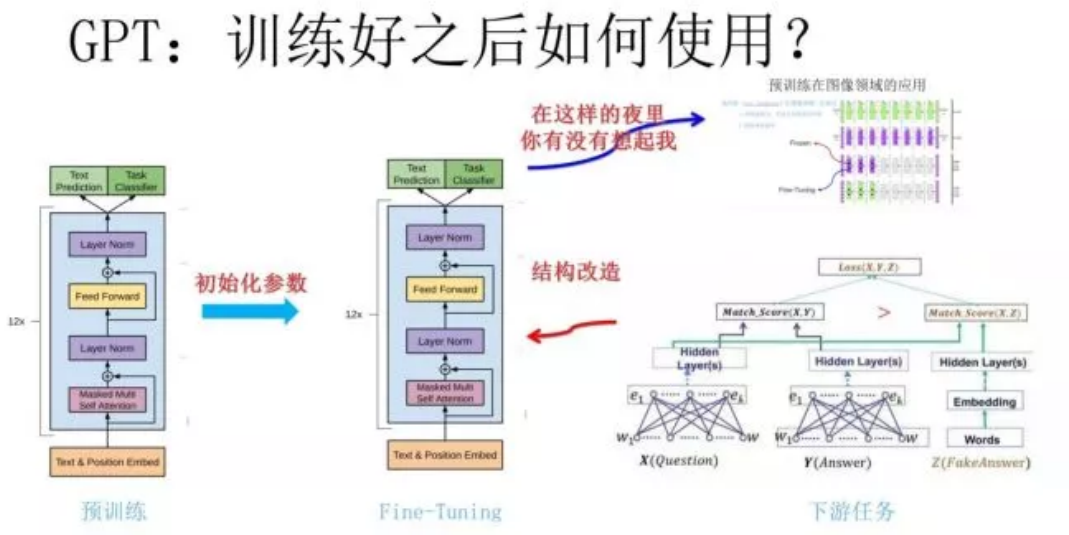


上图展示了 GPT 的预训练过程，其实和 ELMO 是类似的，主要不同在于两点：

1. 特征抽取器不是用的 RNN，而是用的 Transformer，上面提到过它的特征抽取能力要强于 RNN，这个选择很明显是很明智的；
2. ELMO使用上下文对单词进行预测，而 GPT 则只采用 Context-before 这个单词的上文来进行预测，而抛开了下文。

FineTuning

上面讲的是 GPT 如何进行第一阶段的预训练，那么假设预训练好了网络模型，后面下游任务怎么用？它有自己的个性，和 ELMO 的方式大有不同。



在做下游任务的时候，利用第一步预训练好的参数初始化 GPT 的网络结构，这样通过预训练学到的语言学知识就被引入到你手头的任务里来了，这是个非常好的事情。再次，你可以用手头的任务去训练这个网络，对网络参数进行 Fine-tuning，使得这个网络更适合解决手头的问题。

如何改造

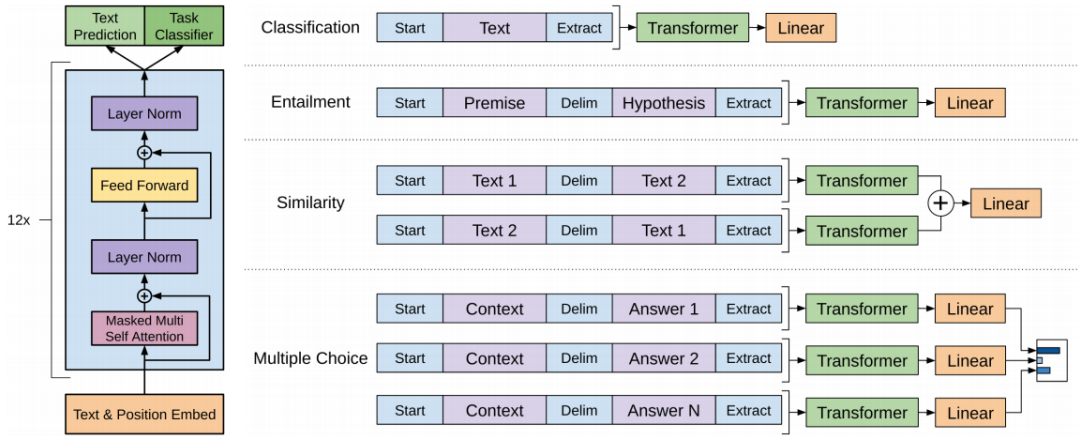


Figure 1: **(left)** Transformer architecture and training objectives used in this work. **(right)** Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear+softmax layer.