第 7 章　自监督与大语言模型

通用人工智能的第一缕曙光

2022 年 11 月 30 日，美国人工智能研究公司 OpenAI 发布 ChatGPT，仅仅两个月，用户数量就已突破了 1 亿，成为有史以来用户增长速度最快的消费级应用程序。那么，ChatGPT 究竟是什么，为什么他有如此大的魅力，他背后的原理究竟是什么。本章将以这些问题为线索，围绕自监督与大语言模型，展开讨论。

正如 1.3 节谈到的，大语言模型源于 2017 年谷歌（Google）公司提出的 Transformer 这种新型神经网络。基于此，谷歌和 OpenAI 分别发展出 Bert 和 GPT 两条主要技术路线。ChatGPT 及其基础模型 GPT-3.5，综合运用了自监督、有监督、环境监督和强化学习等学习范式。本章侧重探讨自监督，下一章则侧重探讨环境监督和强化学习。

7.1　Transformer

回顾 2.5 节探讨的全连接多层神经网络，为了解决 5 万个手写数字图像样本的训练问题，定义了一个 3 层的神经网络，各层的神经元个数分别为 784、30 和 10，因此总的参数量为 784x30+30+30x10+10=23860。用 5 万除以总的参数量，得到每个参数表达约 2 个样本。这个结果表明，全连接网络的参数量非常大，随着层数的增加很容易导致模型过拟合。另外，全连接网络丢弃了手写数字图像的 2 维空间结构，当成 1 维向量进行处理，这一做法显然也是不合理的。围绕这两方面的问题，各种“非全连接神经网络”被陆续提了出来，包括著名的卷积神经网络和近年来影响巨大的 Transformer。

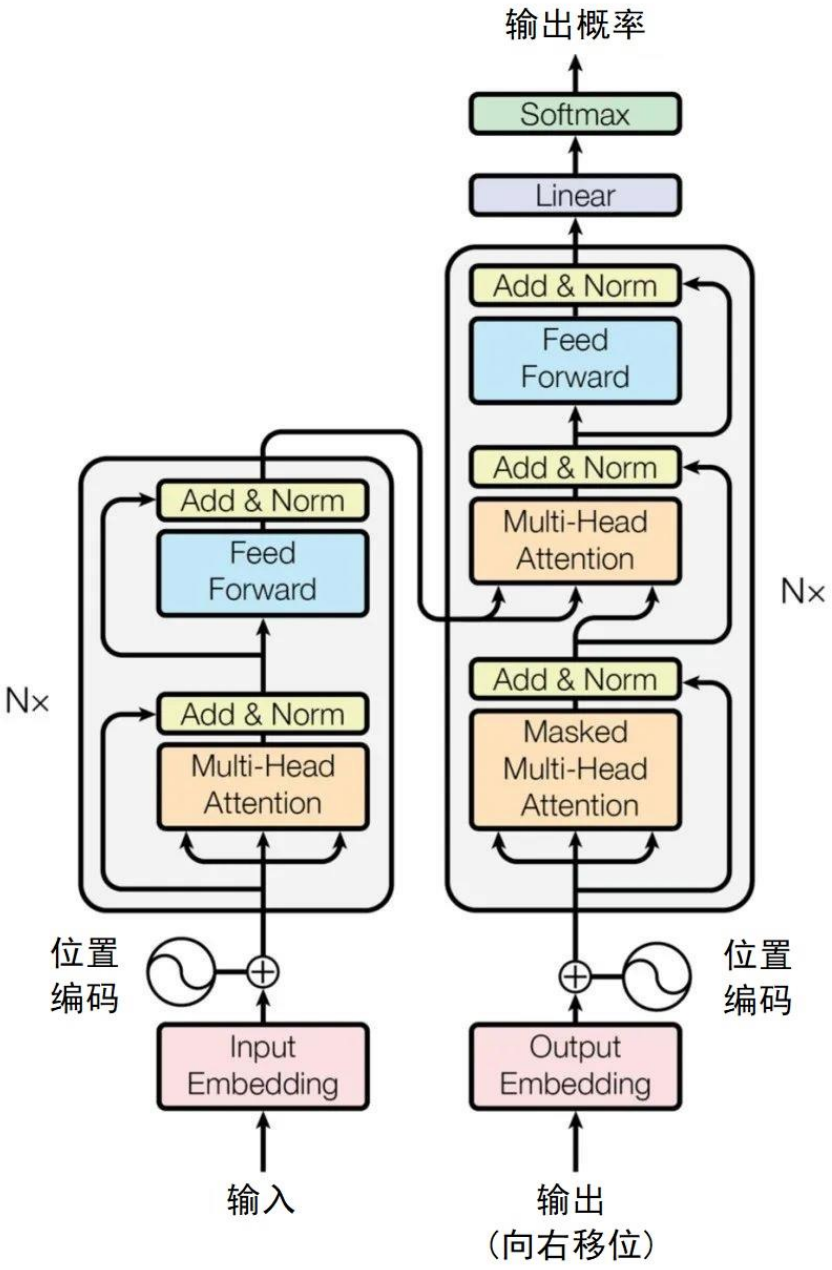


图 7.1 Transformer 整体结构

如图 7.1 所示，Transformer 由左边的编码器和右边的解码器两部分构成。而编码器和解码器都通过 Transformer 层堆叠而成。如图 7.2 所示，Transformer 层由自注意力、全连接网络（前馈网络）和残差连接组成。全连接网络在 2.5 节介绍过了，下面先简单介绍下残差连接，然后重点介绍自注意力。

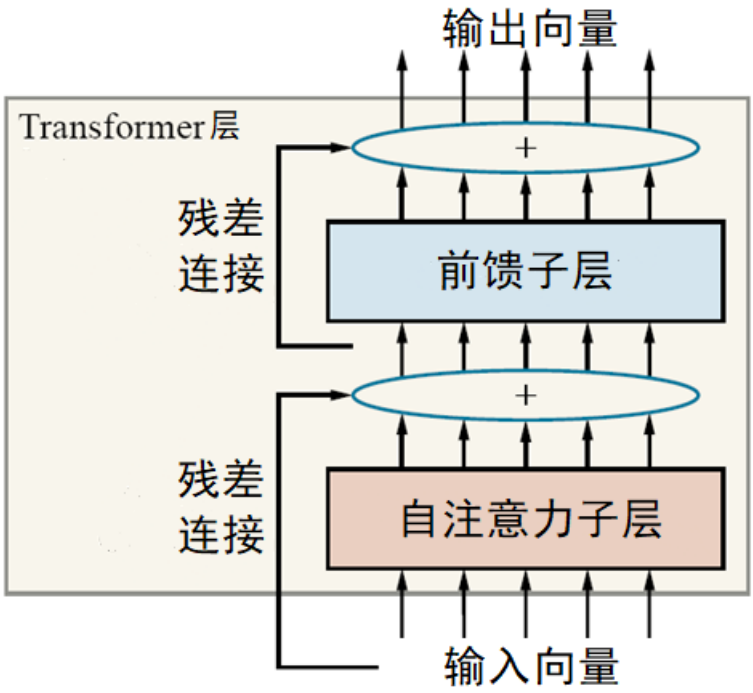


图 7.2 Transformer 层

所谓残差连接，其实就是通过“跳跃连接”来消除深度，从而从根本上解决深度网络难以训练的难题。正如 1.3 节谈到的，残差连接已经成为了现代神经网络的标配，其影响具有基础性。

7.1.1 自注意力

简单来说，所谓“自注意力”就是在没有顺序依赖的情况下对远距离上下文进行建模。

回顾 6.4 节，无论是词集还是词袋，都假定一个句子中词与词之间没有相关性，相互之间独立。基于此假定建立的朴素贝叶斯分类器，在文本分类任务上也能获得相当好的性能。但是，很显然，一个句子中词与词之间是有相关关系或者上下文关系的，因此这个假定显然是错误的！如果基于此假定来生成文本，将导致无法生成有意义的、连贯的句子。

自注意力采用了创新的方式，来对一段文本（不限于一个句子）中的长距离上下文进行建模。具体来讲，自注意力首先使用 3 个不同的权重矩阵，将输入词向量$\boldsymbol{x}_i$投影到 3 种不同的表示中[1]。

第一种表示称为“查询向量”，定义为：

$$\boldsymbol{q}_i = \boldsymbol{W}_q \boldsymbol{x}_i \tag{7.1}$$

是注意力所来自的对象。

第二种表示称为“键向量”，定义为：

$$\boldsymbol{k}_i = \boldsymbol{W}_k \boldsymbol{x}_i \tag{7.2}$$

是注意力所去到的对象。

第三种表示称为“值向量”，定义为：

$$\boldsymbol{v}_i = \boldsymbol{W}_v \boldsymbol{x}_i \tag{7.3}$$

是正在生成的上下文。

由此，第i个词$\boldsymbol{x}_i$的编码结果$\boldsymbol{c}_i$就可以通过对投影向量应用注意力机制来计算：

$$r_{ij} = (\boldsymbol{q}_i \cdot \boldsymbol{k}_j)/\sqrt{d} \tag{7.4}$$

$$a_{ij} = e^{r_{ij}} / \sum_k e^{r_{ik}} \tag{7.5}$$

注 1：两个矩阵相乘就是多组向量之间的内积，每个内积就是一个向量到另一个向量的投影，目的是度量这两个向量的相似性或相关程度。

$$\boldsymbol{c}_i = \sum_j a_{ij}\boldsymbol{v}_j \tag{7.6}$$

其中，d是$\boldsymbol{k}$和$\boldsymbol{q}$的维数。注意，索引值i和j一般是针对同一段文本而言，$\boldsymbol{x}_i$是需要查询注意力的词（注意力所来自的对象，即目标词），而$\boldsymbol{x}_j$是这段文本中被查询的其他某个词（注意力所去到的对象，即源词）。整体来看，式（7.4）通过$\boldsymbol{q}_i$与$\boldsymbol{k}_j$的内积计算$\boldsymbol{x}_i$和$\boldsymbol{x}_j$的相关程度；然后式（7.5）将各词之间的相关程度进行归一化，得到归一化的相关系数；最后式（7.6）针对上下文应用相关系数进行加权求和，最终得到词$\boldsymbol{x}_i$相对其他词$\boldsymbol{x}_j$的注意力编码结果$\boldsymbol{c}_i$。

有几点需要注意。首先，r_{ij}和r_{ji}不同，这意味着自注意力是不对称的，词$\boldsymbol{x}_i$到$\boldsymbol{x}_j$的自注意力不同于$\boldsymbol{x}_j$到$\boldsymbol{x}_i$的自注意力。其次，比例因子$\sqrt{d}$可以提高数值稳定性。再者，一段文本中所有词的编码可以同时计算，这意味着可以采用大规模并行计算进行高效的加速。

最重要的一点是，自注意力的 3 个权重矩阵$\boldsymbol{W}_q$、$\boldsymbol{W}_k$和$\boldsymbol{W}_v$都是从训练样本中学习来的。对于一段文本，第i个词$\boldsymbol{x}_i$的编码结果$\boldsymbol{c}_i$，实际上就是其前面所有词构成的上下文的概要——基于上下文的概要。对于同一段文本，可以学习多组自注意力，以捕捉不同的上下文关系，然后将他们拼接起来，这就是所谓的“多头自注意力（Multi-Head Attention）”。采用“拼接”而非“求和”，有利于尽可能保留丰富的上下文信息。

自注意力、全连接网络（前馈网络）和残差连接组成了 Transformer 层，一个实用的 Transformer 模型通常由 6 个或更多 Transformer 层堆叠而成。

7.1.2 词嵌入

图 7.1 中的输入，即词向量$\boldsymbol{x}_i$，采用的是词嵌入（Embedding）。那么，什么是词嵌入呢？

首先来回顾 6.4 节采用的基于词集或词袋创建的文本向量。以词集为例，文本向量（初值为 0）的长度为词集的长度，如果输入文本里出现词集里的词，则将文本向量对应元素置为 1；否则，仍为初值 0。这样，文本向量就反映了词集里的词在输入文本里出现的情况，0 表示未出现，1 表示出现。

读者可以思考下，这种表示方式存在什么弊端？第一个明显的弊端是：维数高，词集有多大，向量的维数就有多高。第二点，用这种方式表示的一个词（对应一个独热向量）或一段文本（对应一个非独热向量）不能反映出相互之间的相似性，比如词性的相似性、语义的相似性等。

为了克服这两个弊端，词嵌入表示方式被提了出来。其核心思想是，从数据中学习词的低维向量表示，这种表示同时能够捕捉词之间的相似性，即相似的词距离比较近、不相似的词距离比较远。已经有一些比较常用的通用预训练词嵌入词典可供使用，比如 WORD2VEC、GloVe 等，这些词典都是在大规模通用语言数据上，采用无监督或自监督学习方式训练出来的。也可以针对特定任务，采用有监督、无监督或自监督的方式训练特定的词典。

1.1.3 节简单介绍了“自监督学习”的概念。本节以 GloVe（全局向量）词典为例，介绍其采用的一种自监督学习方式。

首先定义一个文本上的滑动窗口（比如窗口大小为 5，就意味着其包含 5 个词）。定义X_{ij}为词i和j在一个窗口内同时出现的次数，X_i为词i与其他任何词同时出现的次数，则$p_{ij} = X_{ij}/X_i$为词j在词i的上下文中出现的概率，称为“共现概率”。

GloVe 的基本思想是，将给定的两个词（比如“冰”和“水蒸气”）与其他词进行比较，以充分捕捉这两个词之间的关系：

$$p_{w,\text{冰}}/p_{w,\text{水蒸气}} \tag{7.7}$$

其中，w表示其他词。这个式子是“冰”和“水蒸气”与w的共现概率之比。比如，w为“固体”这个词时，共现概率比就较高；而w为“气体”这个词时，共现概率比则较低；而w为“水”这个词时，由于与两者都同样相关，共现概率比将接近 1；而如果w为“时尚”这个

词时，由于与两者都同样不相关，共现概率比同样将接近 1。

基于这个基本思想，GloVe 最终将两个词嵌入向量的点积转化为共现概率的对数（越相似共现概率越大），并进一步得到损失函数——一个加权的最小二乘。

7.1.3 位置编码

由 7.1.1 节可知，自注意力与文本中词的顺序无关，因此需要进一步考虑如何将词在文本序列中的相对或绝对位置加入进来。Transformer 采用的是基于不同频率正余弦函数的位置编码（Positional Encoding）：

$$PE_{(pos,2\mathrm{i})} = \sin\left(\frac{pos}{10000^{2i/d_m}}\right) \tag{7.8}$$

$$PE_{(pos,2\mathrm{i}+1)} = \cos\left(\frac{pos}{10000^{2i/d_m}}\right) \tag{7.9}$$

其中，d_m=512，是词嵌入向量的维数；pos是词在文本序列中的位置（比如第一个词的 pos 为 0）；i对应PE的维度，其取值范围为$[0,\dots,d_m/2)$，由此可得到各PE_{pos}向量，其维数为d_m，与词嵌入向量一致，比如$PE_{pos=0}$序列为$[0,1,0,1,\dots]$，$PE_{pos=1}$序列为$[\sin\left(\frac{1}{10000^{0/d_m}}\right),\cos\left(\frac{1}{10000^{0/d_m}}\right),\sin\left(\frac{1}{10000^{2/d_m}}\right),\cos\left(\frac{1}{10000^{2/d_m}}\right),\dots]$。

由此，每个位置都有一个唯一的位置编码PE_{pos}。这个位置编码还有一些重要特点：第一，能够适应比训练集里面所有文本序列更长的序列，因为采用的是正余弦函数来进行计算。第二，便于模型学习到词间基于相对位置的相关关系，因为对于相对偏移k，PE_{pos+k}可以通过三角函数公式表示为PE_{pos}的线性函数。

如图 7.1 所示，词嵌入向量和位置编码相加，实现语义信息和位置信息的融合，作为 Transformer 层（图 7.2 所示）的输入。读到这里，读者可能会问一个问题，为什么是相加，而不是类似“多头自注意力”采用拼接呢？相加的话不就把两种信息混淆在一起了吗？可以这样来理解，如果直接拼接的话固然可以，但是代价也很大，维数翻倍。那么，如果相加的话有没可能达到相同的效果呢？答案是可以，因为两个向量相加等同于两个输入向量的拼接再作一个线性变换。

7.1.4 编码器和解码器

前面已经讲到，Transformer 由左边的编码器和右边的解码器两部分构成（图 7.1 所示）。无论是编码器和解码器，都通过 Transformer 层堆叠而成，结构上几乎是相同的。下面以机器翻译任务为例（比如英文翻译为中文，见图 7.3 给出的一个例子），重点注意两者的不同之处。

首先来看编码器（图 7.1 中左边部分）。如图 7.3 所示，输入英文单词“Welcome”，经过词嵌入和位置编码得到输入向量$\boldsymbol{x}_i$，计算其与英文文本序列（“Welcome teacher Li”）其他单词的多头自注意力，然后经过全连接层，最后得到编码器的输出。特别注意：编码器的输出连接到解码器（图 7.1 中右边部分）的中间。

再来看解码器。除了编码器的输出连接到解码器中间这个输入，已经完成翻译的汉语词语（比如“李”）也作为解码器的输入，并计算其与已经完成翻译的汉语文本序列（比如“欢迎李”）其他词语的多头自注意力。接下来，编码器的输出与解码器的输入计算多头注意力（比如“teacher Li”应该对应“李老师”，而不是“老师李”）。最后经过全连接层和 Softmax，输出翻译结果。

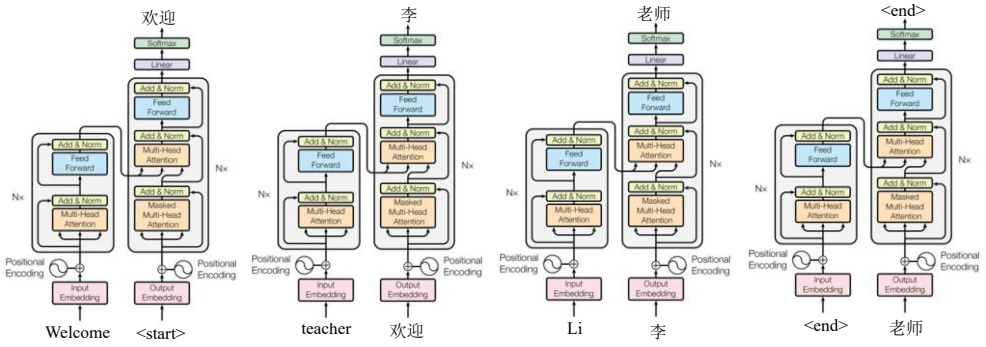


图 7.3 一个英翻汉的例子

7.2 GPT 与大语言模型的预训练

7.1 节介绍的 Transformer 针对的是“机器翻译”这个自然语言应用任务，其具有编码器加解码器结构，编码器对应源语言（比如英语），而解码器对应目标语言（比如中文）。本节将介绍的 GPT（Generative Pretrained Transformer，生成式预训练 Transformer）是 Transformer 的一个变种，其只保留了解码器部分，主要针对的是大语言模型（Large Language Model）的自监督预训练任务。

为什么称为预训练呢？预训练实际上是迁移学习中的一个基本概念，简单说来，就是希望在一个较大的、通用的数据集上学习的预训练模型，能够以较小的代价（比如在一个较小的、专用的数据集上进行精调）应用到不同的下游任务中。GPT 就是一个预训练的大语言模型，能够根据给定的一段文本进行续写，即“生成”合乎语法且语义上连贯的后续文本。有了 GPT，通过迁移学习，就可将其应用到问答系统（比如 ChatGPT）、文本分类（比如情感分析）等诸多下游任务中。7.1.2 节介绍的通用预训练词嵌入词典也是预训练语言模型的一个例子。

那么，GPT 所采用的自监督学习方式具体是怎样的呢？为了说明这一点，首先引入“n 元词模型”这个一般概念：

$$p(w_j|w_{1:j-1}) = p(w_j|w_{j-n+1:j-1}) \tag{7.10}$$

$$p(w_{1:N}) = \prod_{j=1}^{N} p(w_j|w_{j-n+1:j-1}) \tag{7.11}$$

如式（7.10）所示，一般而言，一段文本的第j个词w_j依赖于其前面所有$j-1$个词，“n 元词模型”将其简化为w_j仅依赖于其前面$n-1$个词（$n \le j$）。由此，如式（7.11）所示，一段长度为N的文本，其联合概率分布$p(w_{1:N})$就可以按照“n 元词模型”进行简化：每个词的条件概率的乘积。

取 n=1，可得到：

$$p(w_j|w_{1:j-1}) = p(w_j|w_{j:j-1}) = p(w_j) \tag{7.12}$$

$$p(w_{1:N}) = \prod_{j=1}^{N} p(w_j) \tag{7.13}$$

这就是 6.4 节介绍的词集或词袋（即朴素贝叶斯分类器在文本分类上的应用），一个句子中词与词之间没有相关性，相互独立。

取 n=2，可得到：

$$p(w_j|w_{1:j-1}) = p(w_j|w_{j-1}) \tag{7.14}$$

$$p(w_{1:N}) = \prod_{j=1}^{N} p(w_j|w_{j-1}) \tag{7.15}$$

这就类似 6.3 节介绍的，一种常见的半朴素贝叶斯分类器：每个特征在类别之外仅依赖于一

个其他特征。

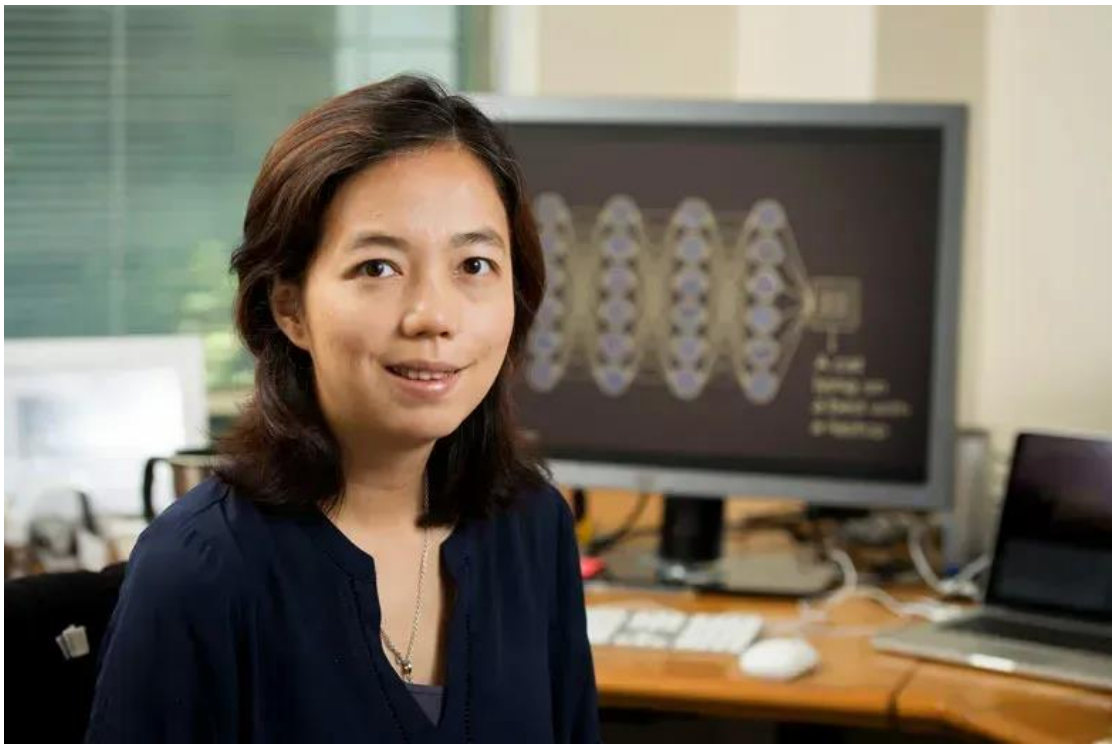
另外，7.1.2 节介绍的 GloVe 词典采用文本上的滑动窗口，这个滑动窗口的大小本质上就是指的“n 元词模型”中的 n。

至此，具体到 GPT，其实采用的就是一个“n 元词模型” $p(w_j|w_{j-n+1:j-1})$，目标是基于前面 n-1 个词$w_{j-n+1:j-1}$，预测下一个（即第 n 个）词w_j。n 就是上下文长度。

GPT 采用了约 10 亿词的训练数据进行自监督预训练，采用了 12 层仅有解码器部分的 Transformer，自注意力为 12 头，输入词嵌入向量的维数为 768，总参数量达到 1.17 亿。后续，GPT-2 的层数达到了 48 层，输入词嵌入向量的维数为 1600，总参数量达到 15.42 亿。用于自监督预训练的文本数据则达到了 40GB。GPT-3 用于自监督预训练的文本数据达到了 570GB，总参数量为 1750 亿。GPT-3.5 则进一步用 179GB 来自 GitHub 上的代码进行了自监督预训练。GPT-4 据说总参数量达到了 1.8 万亿。

就上下文长度而言，GPT 为 512，GPT2 增加到了 1024，GPT3 则达到了 2048。

小故事：李飞飞与 ImageNet



在 AI 领域，华人也发挥着巨大影响力。2024 年 2 月 24 日，芯片巨头英伟达宣布成立一个新研究部门——通用具身智能体研究实验室。该实验室的领导者是两位华人 90 后博士——范麟熙（Jim Fan）和朱玉可（Yuke Zhu）。而这两人的导师，则更为重要——被称为“AI 教母”的华人科学家李飞飞。

为了赚钱，李飞飞在饭店刷过盘子，在干洗店打过工，整个高中和大学时代，她的衣服都是从别人丢掉的垃圾中捡的。

没人想到，这样一个贫穷的女孩，之后会成为席卷全球的 AI 革命的核心人物之一，甚至被誉为“AI 教母”。

她的征途是星辰和大海，起点却是美国东北部的臭水沟。

16 岁时，李飞飞和父母移民到美国，生活跌入谷底：一家三口挤在一个只有一间卧室的公寓里，没有积蓄，不会说英语，靠繁重的体力劳动维持生计。

这个聪明的女孩考上美国最顶尖的高校之一，却没想着毕业后挣大钱实现阶层跃迁，而

是投入到当时还是“天坑”的人工智能专业中，梦想着教会机器学习，改变整个世界。

坚信数据对人工智能有重要意义的她，在只有一个助手的条件下，创建了人类历史上规模最大的计算机视觉标注数据集 ImageNet，为接下来引爆 AI 革命的大事件准备了舞台。可以说，没有 ImageNet 这个关键催化剂，就没有现在的深度学习和 AI 革命。

李飞飞一路走来，生在北京，长于四川，又从中国到美国，由物理专业到人工智能领域，靠着其坚强、疯狂与热爱，从一个洗衣妹跨进 AI 这场科技革命的中心。

启迪：

（1）科研有时候就如同一场“豪赌”，认准了就要大胆地“赌一把”。

（2）找到属于自己的那颗“北极星”。

（3）王国维的三重境界说：“昨夜西风凋碧树，独上高楼，望尽天涯路。”此第一境界也（远大的**目标**）。“衣带渐宽终不悔，为伊消得人憔悴。”此第二境界也（**坚持**）。“众里寻他千百度，蓦然回首，那人却在，灯火阑珊处。”此第三境界也（机缘巧合、水到渠成的**心态**）。