ChatGPT的工作原理

— 0 —

前言

ChatGPT 能够自动生成一些读起来表面上甚至像人写的文字的东西，这非常了不起，而且出乎意料。但它是如何做到的？为什么它能发挥作用？我在这里的目的是大致介绍一下 ChatGPT 内部的情况，然后探讨一下为什么它能很好地生成我们认为是有意义的文本。

还没使用过ChatGPT的伙伴可以点击下面链接直接使用（不需要科学上网工具，后台对接的是OpenAI和微软的官方接口）：
https://chatgpt.zntjxt.cn/

我首先要说明一下，我将把重点放在正在发生的事情的大的方向上，虽然我会提到一些工程细节，但我不会深入研究它们。（我所说的实质内容也同样适用于目前其他的“大型语言模型”LLM 和 ChatGPT）。

首先要解释的是，ChatGPT 从根本上说总是试图对它目前得到的任何文本进行“合理的延续”，这里的“合理”是指“在看到人们在数十亿个网页上所写的东西之后，人们可能会期望某人写出什么”。

资料分享群

学场圈

你想要资料这里都有

1.每日群内分享10+最新【行业报告】
2.各行业最新顶级【策划方案】
3.日更【电子书】【音频书】【读书笔记】
4.不定时分享最新【资讯】
5.国内外报刊【华尔街日报】【金融时报】【读者】【财经】
6.所有资料均为公开版，版权归作者所有，学场只做内部学习。

创业 投资 学习 职场 资源

扫码进群

高价值内容分享圈

因此，假设我们已经得到了 “人工智能最好的是它能去做 ……” 的文本（“The best thing about AI is its ability to”）。想象一下，扫描数十亿页的人类书写的文本（例如在网络上和数字化书籍中），并找到这个文本的所有实例 —— 然后看到什么词在接下来的时间里出现了多少。

ChatGPT 有效地做了类似的事情，除了（正如我将解释的）它不看字面文本；它寻找在某种意义上 “意义匹配” 的东西。但最终的结果是，它产生了一个可能出现在后面的词的排序列表，以及 “概率”。

The best thing about AI is its ability to

learn	4.5%
predict	3.5%
make	3.2%
understand	3.1%
do	2.9%

值得注意的是，当 ChatGPT 做一些事情，比如写一篇文章时，它所做的基本上只是反复询问 “鉴于到目前为止的文本，下一个词应该是什么？” —— 而且每次都增加一个词。（更准确地说，正如我将解释的那样，它在添加一个 “标记”，这可能只是一个词的一部分，这就是为什么它有时可以 “编造新词”）。

在每一步，它得到一个带有概率的单词列表。但是，它究竟应该选择哪一个来添加到它正在写的文章（或其他什么）中呢？人们可能认为它应该是 “排名最高” 的词（即被分配到最高 “概率” 的那个）。

但是，这时就会有一点巫术开始悄悄出现。因为出于某种原因 —— 也许有一天我们会有一个科学式的理解 —— 如果我们总是挑选排名最高的词，我们通常会得到一篇非常“平淡”的文章，似乎从来没有“显示出任何创造力”（甚至有时一字不差地重复）。但是，如果有时（随机的）我们挑选排名较低的词，我们会得到一篇“更有趣”的文章。

这里有随机性的事实意味着，假如我们多次使用同一个提示，我们也很可能每次都得到不同的文章。而且，为了与巫术的想法保持一致，有一个特定的所谓“温度”参数（temperature parameter），它决定了以什么样的频率使用排名较低的词，而对于论文的生成，事实证明，0.8 的“温度”似乎是最好的。（值得强调的是，这里没有使用任何“理论”；这只是一个在实践中被发现可行的问题）。例如，“温度”的概念之所以存在，是因为恰好使用了统计物理学中熟悉的指数分布，但没有“物理”联系 —— 至少到目前为止我们如此认为。）

在我们继续之前，我应该解释一下，为了论述的目的，我大多不会使用 ChatGPT 中的完整系统；相反，我通常会使用更简单的 GPT-2 系统，它有一个很好的特点，即它足够小，可以在标准的台式电脑上运行。

因此，对于我展示的所有内容，包括明确的沃尔弗拉姆语言（Wolfram Language）代码，你可以立即在你的计算机上运行。

例如，这里是如何获得上述概率表的。首先，我们必须检索底层的“语言模型”神经网：

```
In[ ]:= model =
          NetModel[{"GPT2 Transformer Trained on WebText Data",
            "Task" → "LanguageModeling"}]

Out[ ]= NetChain[ Input port:  string
                  Output port: class ]
```

稍后，我们将看看这个神经网的内部，并谈谈它是如何工作的。但现在我们可以把这个“网络模型”作为一个黑匣子应用于我们迄今为止的文本，并要求按概率计算出该模型认为应该选择的前五个词：

```
In[ ]:= model["The best thing about AI is its ability to", {"TopProbabilities", 5}]

Out[ ]= {do → 0.0288508, understand → 0.0307805,
         make → 0.0319072, predict → 0.0349748, learn → 0.0445305}
```

这就把这个结果变成了一个明确的格式化的“数据集”：

```
In[ ]:= Dataset[ReverseSort[Association[%]],
          ItemDisplayFunction → (PercentForm[#, 2] &)]
```

Out[ ]=

learn	4.5%
predict	3.5%
make	3.2%
understand	3.1%
do	2.9%

如果重复“应用模型”——在每一步中加入概率最高的词（在此代码中被指定为模型的“决定”），会发生什么：

```
In[•]:= NestList[StringJoin[#, model[#, "Decision"]] &,
          "The best thing about AI is its ability to", 7]

Out[•]= {The best thing about AI is its ability to,
         The best thing about AI is its ability to learn,
         The best thing about AI is its ability to learn from,
         The best thing about AI is its ability to learn from experience,
         The best thing about AI is its ability to learn from experience.,
         The best thing about AI is its ability to learn from experience. It,
         The best thing about AI is its ability to learn from experience. It's,
         The best thing about AI is its ability to learn from experience. It's not}
```

如果再继续下去会发生什么？在这种情况下（“零温度”），很快就会出现相当混乱和重复的情况：

```
The best thing about AI is its ability to learn from experience. It's not just a matter
of learning from experience, it's learning from the world around you. The AI is a very
good example of this. It's a very good example of how to use AI to improve your life.
It's a very good example of how to use AI to improve your life. The AI is a very good
example of how to use AI to improve your life. It's a very good example of how to use AI to
```

但是，如果不总是挑选“顶级”词，而是有时随机挑选“非顶级”词（“随机性”对应“温度”为 0.8）呢？人们又可以建立起文本：

{The best thing about AI is its ability to,
The best thing about AI is its ability to create,
The best thing about AI is its ability to create worlds,
The best thing about AI is its ability to create worlds that,
The best thing about AI is its ability to create worlds that are,
The best thing about AI is its ability to create worlds that are both,
The best thing about AI is its ability to create worlds that are both exciting,
The best thing about AI is its ability to create worlds that are both exciting,}

而每次这样做，都会有不同的随机选择，文本也会不同 —— 如这 5 个例子：

The best thing about AI is its ability to learn. I've always liked the

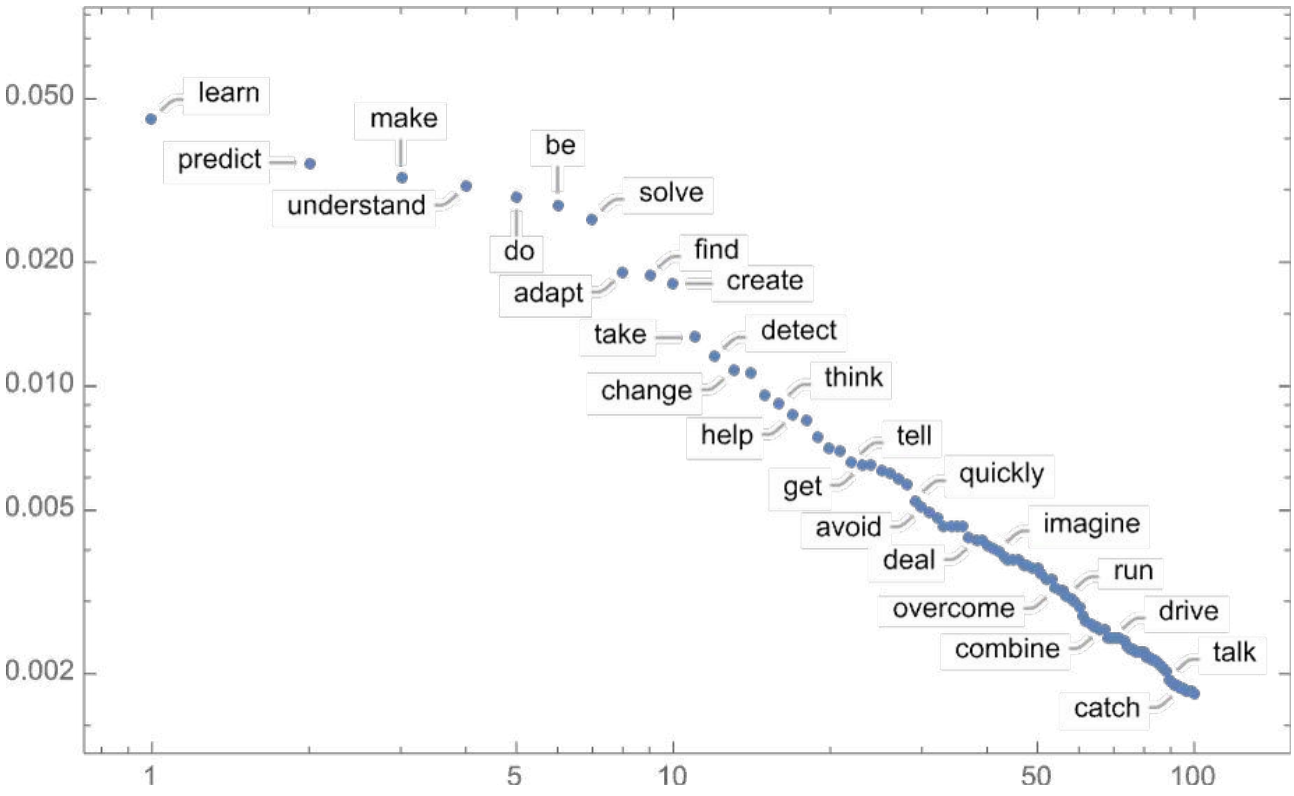
The best thing about AI is its ability to really come into your world and just

The best thing about AI is its ability to examine human behavior and the way it

The best thing about AI is its ability to do a great job of teaching us

The best thing about AI is its ability to create real tasks, but you can

值得指出的是，即使在第一步，也有很多可能的 “下一个词” 可供选择（温度为 0.8），尽管它们的概率下降得很快（是的，这个对数图上的直线对应于 n-1 的 “幂律” 衰减，这是语言的一般统计的特点）：



那么，如果继续下去会发生什么？这里有一个随机的例子。它比顶层词（零温度）的情况要好，但顶多还是有点奇怪：

The best thing about AI is its ability to learn from experience. It's not just a matter of learning from experience, it's learning from the world around you. The AI is a very good example of this. It's a very good example of how to use AI to improve your life. It's a very good example of how to use AI to improve your life. The AI is a very good example of how to use AI to improve your life. It's a very good example of how to use AI to

这是用最简单的 GPT-2 模型（来自 2019 年）做的。用较新和较大的 GPT-3 模型，结果更好。这里是用同样的 “提示” 产生的顶部文字（零温度），但用最大的 GPT-3 模型：

The best thing about AI is its ability to automate processes and make decisions quickly and accurately. AI can be used to automate mundane tasks, such as data entry, and can also be used to make complex decisions, such as predicting customer behavior or analyzing large datasets. AI can also be used to improve customer service, as it can quickly and accurately respond to customer inquiries. AI can also be used to improve the accuracy of medical diagnoses and to automate the process of drug discovery.

这是“温度为 0.8”时的一个随机例子：

The best thing about AI is its ability to learn and develop over time, allowing it to continually improve its performance and be more efficient at tasks. AI can also be used to automate mundane tasks, allowing humans to focus on more important tasks. AI can also be used to make decisions and provide insights that would otherwise be impossible for humans to figure out.

— 1 —

概率从何而来?

好吧，ChatGPT 总是根据概率来选择下一个词。但是这些概率从何而来？

让我们从一个更简单的问题开始。让我们考虑一次生成一个字母（而不是单词）的英语文本。我们怎样才能算出每个字母的概率呢？

我们可以做的一个非常简单的事情就是取一个英语文本的样本，然后计算不同字母在其中出现的频率。因此，举例来说，这是计算维基百科上关于“猫”（cat） 的文章中的字母：

```
In[ ]:= LetterCounts[WikipediaData["cats"]]

Out[ ]= <| e → 4279, a → 3442, t → 3397, i → 2739, s → 2615, n → 2464, o → 2426,
  r → 2147, h → 1613, l → 1552, c → 1405, d → 1331, m → 989, u → 916,
  f → 760, g → 745, p → 651, y → 591, b → 511, w → 509, v → 395, k → 212,
  T → 114, x → 85, A → 81, C → 81, I → 68, S → 55, F → 42, z → 38, E → 36
```

而这对“狗”（dog） 也有同样的作用：

```
In[ ]:= LetterCounts[WikipediaData["dogs"]]

Out[ ]= <| e → 3911, a → 2741, o → 2608, i → 2562, t → 2528, s → 2406,
  n → 2340, r → 1866, d → 1584, h → 1463, l → 1355, c → 1083, g → 929,
  m → 859, u → 782, f → 662, p → 636, y → 500, b → 462, w → 409,
  v → 406, k → 151, T → 90, C → 85, I → 80, A → 74, x → 71, S → 65,
```

结果相似，但不一样（“o” 在 “dogs” 文章中无疑更常见，因为毕竟它出现在 “dog” 这个词本身）。尽管如此，如果我们采取足够大的英语文本样本，我们可以期待最终得到至少是相当一致的结果。

```
In[ ]:= English LANGUAGE [ character frequencies ]

Out[ ]= {e → 12.7%, t → 9.06%, a → 8.17%, o → 7.51%, i → 6.97%, n → 6.75%,
  s → 6.33%, h → 6.09%, r → 5.99%, d → 4.25%, l → 4.03%, c → 2.78%, u → 2.76%,
  m → 2.41%, w → 2.36%, f → 2.23%, g → 2.02%, y → 1.97%, p → 1.93%, b → 1.49%,
  v → 0.978%, k → 0.772%, j → 0.153%, x → 0.150%, q → 0.0950%, z → 0.0740%}
```

下面是我们得到的一个样本，如果我们用这些概率生成一个字母序列：

```
rronoitadatcaeaesaotdoysaroiyiinnbantoioestlhddeocneooewceseciselnodrtrdgriscsatsepesdcnio
    uhoetsedeyhedslernevstothindtbmnaohngotannbthrdthtonsipieldn
```

我们可以通过添加空格将其分解为 “单词”，就像它们是具有一定概率的字母一样：

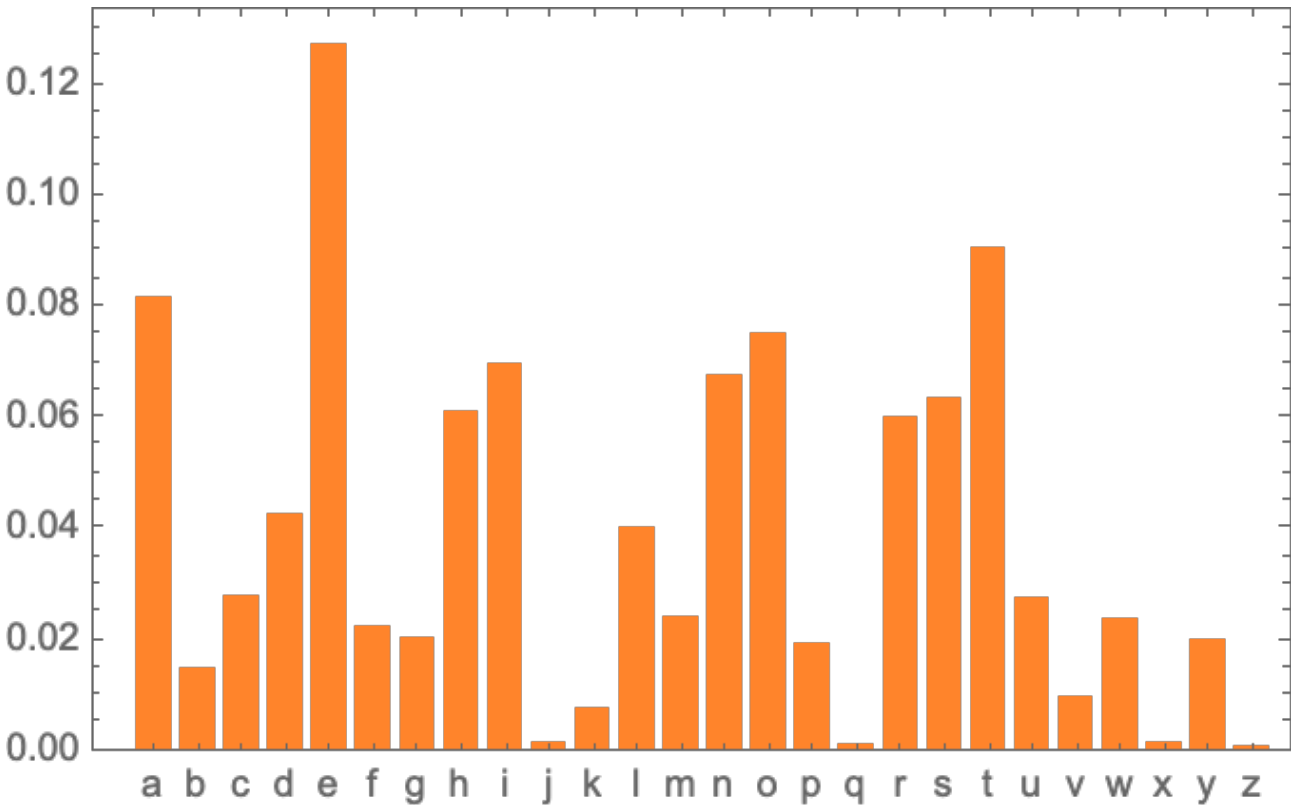
sd n oeiaim satnwhoo eer rtr ofiianordrenapwokom del oaas ill e h f
rellptohltvoettseodtrncilntehtotrkthrslo hdaol n sriaefr hthehtn ld gpod a h y oi

我们可以通过强迫 “字长” 的分布与英语中的分布相一致，在制造 “单词” 方面做得稍微好一点：

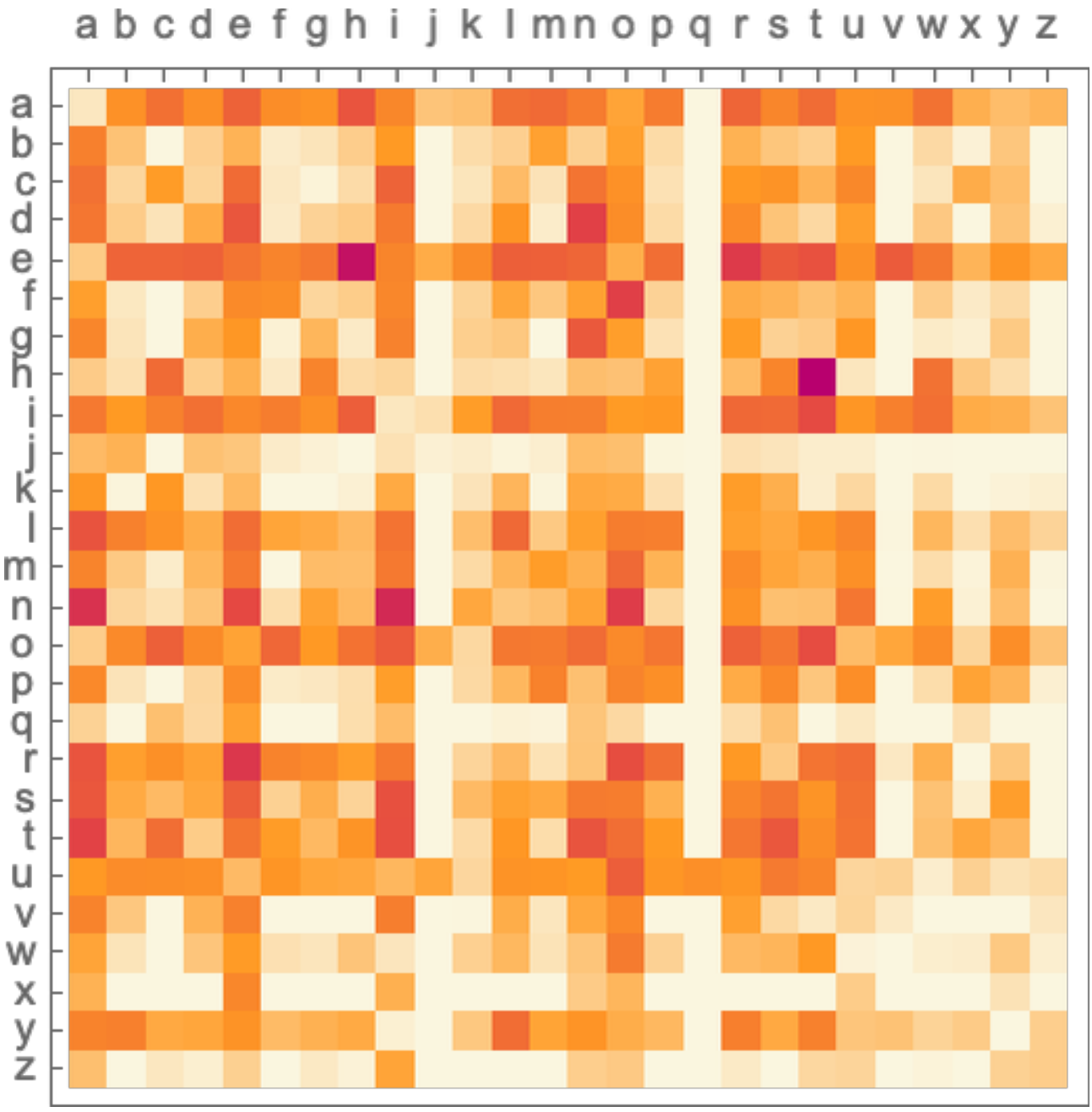
ni hilwhuei kjtn isjd erogofnr n rwhwfao rcuw lis fahte uss cpnc
nlu oe nusaetat llfo oeme rrhrtn xdses ohm oa tne ebedcon oarvthv ist

我们在这里没有碰巧得到任何 “实际的词”，但结果看起来稍好一些。不过，要想更进一步，我们需要做的不仅仅是随机地分别挑选每个字母。例如，我们知道，如果我们有一个 “q”，下一个字母基本上必须是 “u”：

这里有一个字母本身的概率图：



这是一个显示典型英语文本中成对字母（“2-grams”）概率的图。可能的第一个字母显示在页面上，第二个字母显示在页面下：



例如，我们在这里看到，除了“u”行，“q”列是空白的（概率为零）。好了，现在我们不再是一次生成一个字母的“单词”，而是使用这些“2-gram”概率，一次看两个字母来生成它们。下面是一个结果的样本——其中恰好包括一些“实际的词”：

on inguman men ise forerenoft weat iofobato buc ous corew ousesetiv

falle tinouco ryefo ra the ecederi pasuthrgr cuconom tra tesla wil tat pere thi

有了足够多的英语文本，我们不仅可以对单个字母或成对字母（2-grams）的概率进行很好的估计，而且还可以对较长的字母进行估计。如果我们用逐渐变长的 n-gram 概率生成 “随机词”，我们就会发现它们逐渐变得 “更现实”：

0	on gxeeetowmt tsifhy ah aufnsoc ior oia itlt bnc tu ih uls
1	ri io os ot timumumoi gymyestit ate bshe abol viowr wotybeat mecho
2	wore hi usinallistin hia ale warou pothe of premetra bect upo pr
3	qual musin was witherins wil por vie surgedygua was suchinguary outheydays theresist
4	stud made yello adenced through theirs from cent intous wherefo proteined screa
5	special average vocab consumer market prepara injury trade consa usually speci utility

但现在让我们假设 —— 或多或少像 ChatGPT 那样 —— 我们处理的是整个单词，而不是字母。英语中大约有 40,000 个合理的常用词。通过查看大型英语文本语料库（比如几百万本书，总共有几千亿个单词），我们可以得到每个单词的常见程度的估计。利用这一点，我们可以开始生成 “句子”，其中每个词都是独立随机抽取的，其出现的概率与语料库中的相同。下面是我们得到的一个样本：

of program excessive been by was research rate not here of of other is men were against are show they the different the half the the in any were leaved

显然，这是一派胡言。那么，我们如何才能做得更好呢？就像对待字母一样，我们可以开始考虑的不仅仅是单个词的概率，还有成对的或更长的词的 n-grams 的概率。在成对的情况下，以下是我们得到的 5 个例子，所有情况都是从 “猫” 这个词开始的：

cat through shipping variety is made the aid emergency can the

cat for the book flip was generally decided to design of

cat at safety to contain the vicinity coupled between electric public

cat throughout in a confirmation procedure and two were difficult music

cat on the theory an already from a representation before a

它变得稍微“看起来很合理”了。我们可以想象，如果我们能够使用足够长的 n-grams，我们基本上会“得到一个 ChatGPT”—— 在这个意义上，我们会得到一些东西，以“正确的总体论文概率”生成论文长度的单词序列。但问题是：没有足够的英文文本可以推导出这些概率。

在网络的抓取中，可能有几千亿个单词；在已经数字化的书籍中，可能有另外几千亿个单词。但是有了 4 万个常用词，即使是可能的 2-grams 的数量也已经是 16 亿了，可能的 3-grams 的数量是 60 万亿。

所以我们没有办法从现有的文本中估计出所有这些的概率。而当我们达到 20 个字的“文章片段”时，可能性的数量比宇宙中的粒子数量还要多，所以从某种意义上说，它们永远不可能全部被写下来。

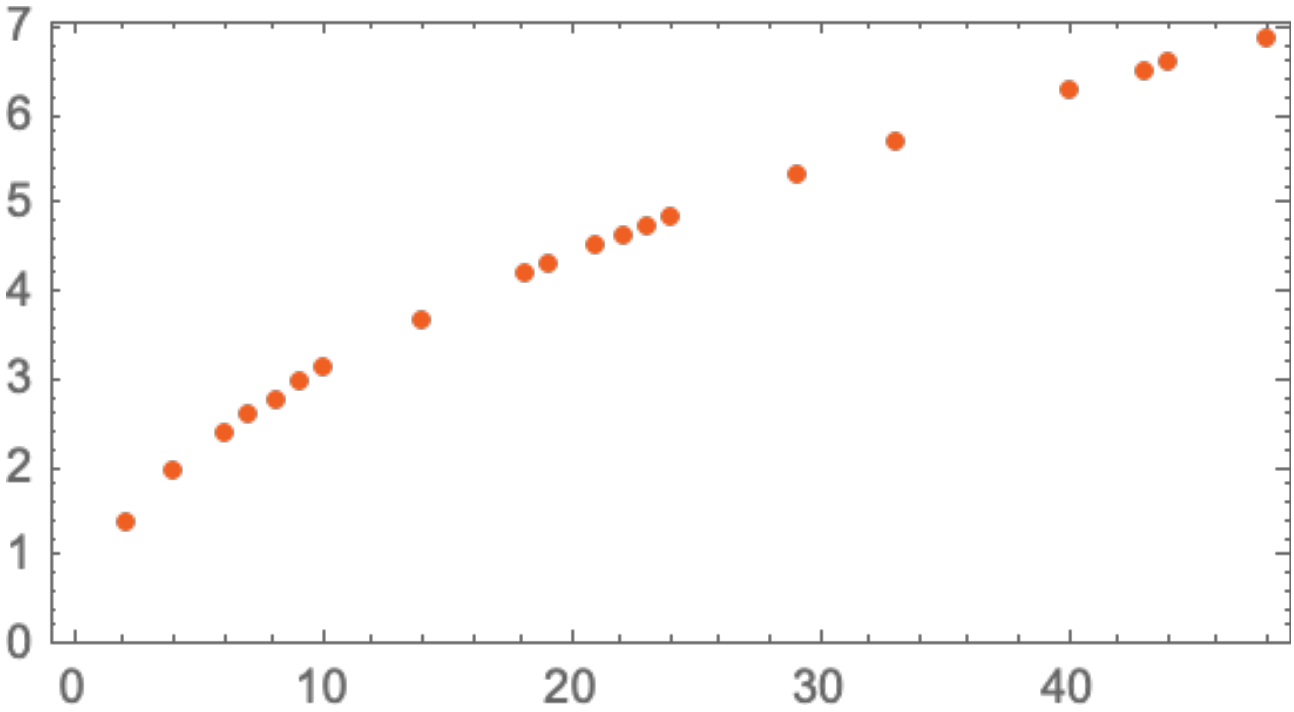
那么我们能做什么呢？最大的想法是建立一个模型，让我们估计序列出现的概率 —— 即使我们在所看的文本语料库中从未明确见过这些序列。而 ChatGPT 的核心正是一个所谓的“大型语言模型”（LLM），它的建立可以很好地估计这些概率。

— 2 —

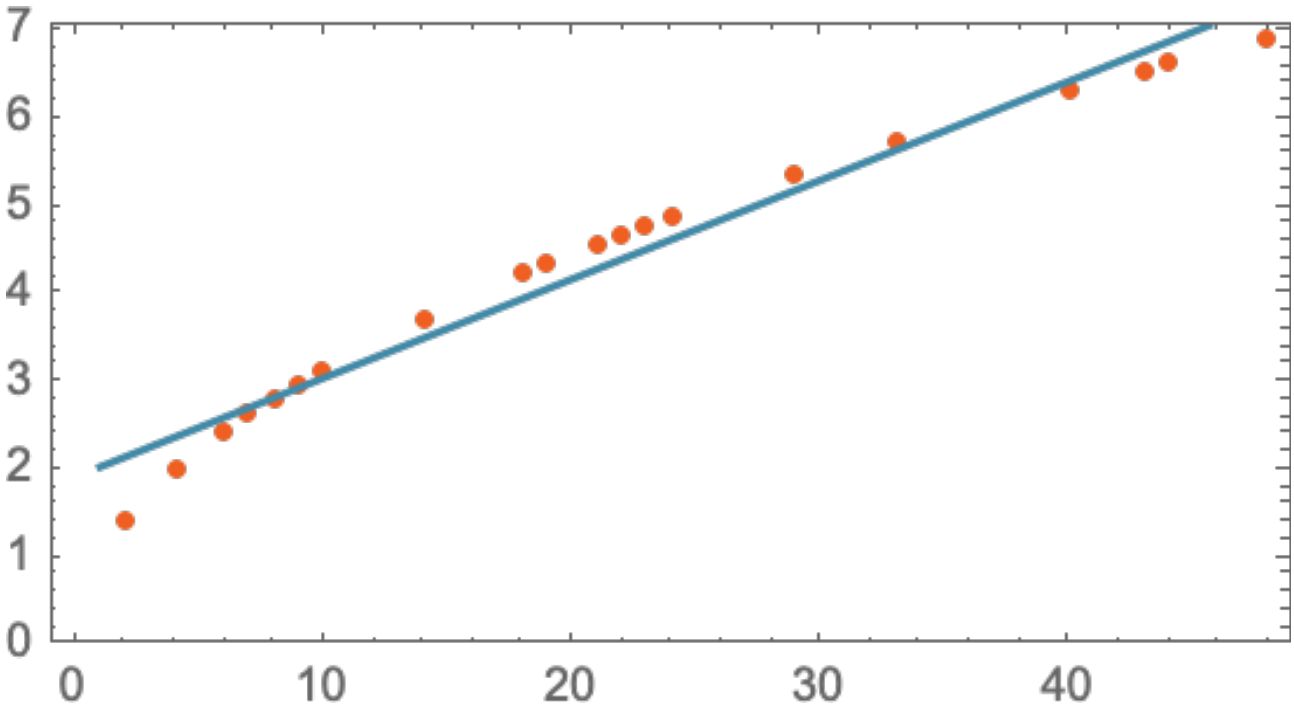
什么是模型?

假设你想知道（就像伽利略在 15 世纪末所做的那样），从比萨塔的每一层落下的炮弹要多长时间才能落地。那么，你可以在每一种情况下测量它，并将结果制成表格。或者你可以做理论科学的精髓：建立一个模型，给出某种计算答案的程序，而不是仅仅测量和记住每个案例。

让我们想象一下，我们有（有点理想化的）数据，说明炮弹从不同楼层落下需要多长时间。

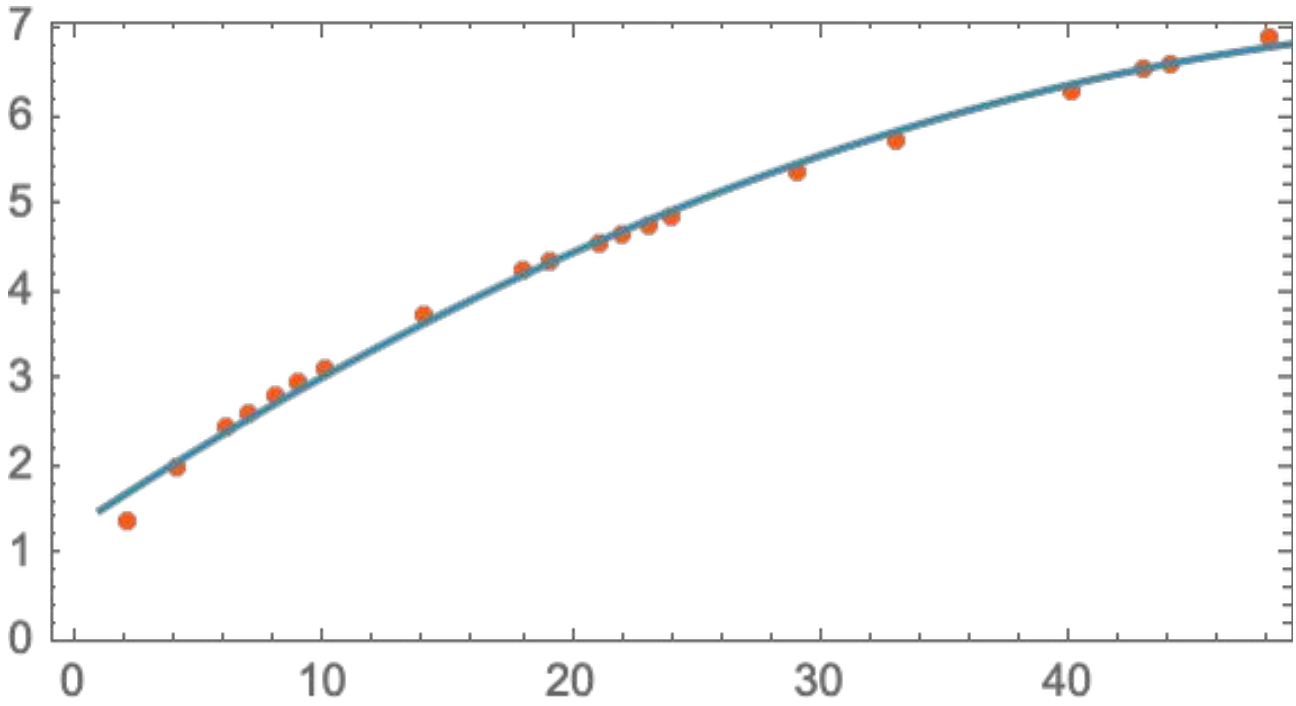


我们如何计算出它从一个我们没有明确数据的楼层落下需要多长时间？在这种特殊情况下，我们可以用已知的物理学定律来计算。但是，如果说我们所得到的只是数据，而我们不知道有什么基本定律在支配它。那么我们可以做一个数学上的猜测，比如说，也许我们应该用一条直线作为模型。

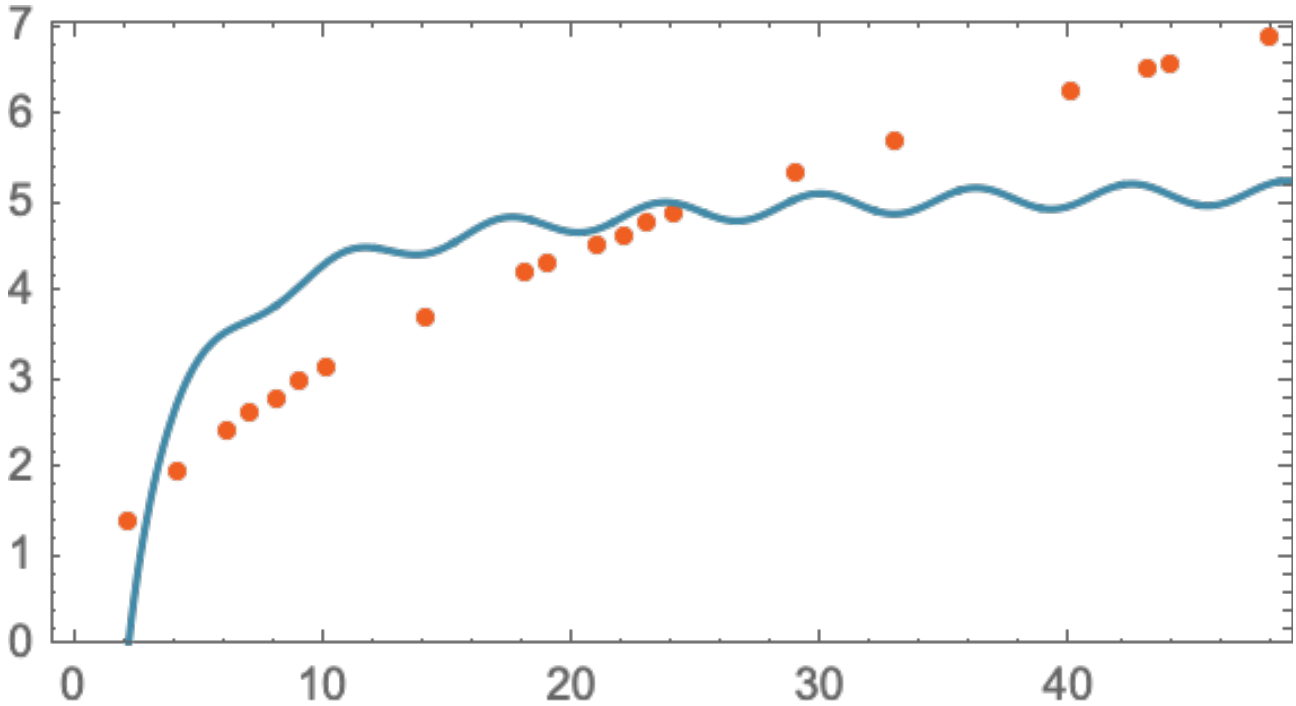


我们可以选择不同的直线。但这是平均来说最接近我们所给的数据的一条。而根据这条直线，我们可以估算出任何楼层的下降时间。

我们怎么知道要在这里尝试使用一条直线呢？在某种程度上我们不知道。这只是数学上简单的东西，而我们已经习惯了这样的事实：我们测量的很多数据都被数学上简单的东西很好地拟合了。我们可以尝试一些数学上更复杂的东西 —— 比如说 a + bx + cx2，然后在这种情况下，我们做得更好：



不过，事情可能会出大问题。比如这里是我们用 a + b/c + x sin(x) 最多也就做成：



值得理解的是，从来没有一个 “无模型的模型”。你使用的任何模型都有一些特定的基础结构，然后有一组 “你可以转动的旋钮”（即你可以设置

的参数）来适应你的数据。而在 ChatGPT 的案例中，使用了很多这样的“旋钮”—— 实际上，有 1750 亿个。

但令人瞩目的是，ChatGPT 的底层结构 —— “仅仅” 有这么多的参数 —— 足以使一个计算下一个单词概率的模型 “足够好”，从而为我们提供合理的文章长度的文本。

— 3 —

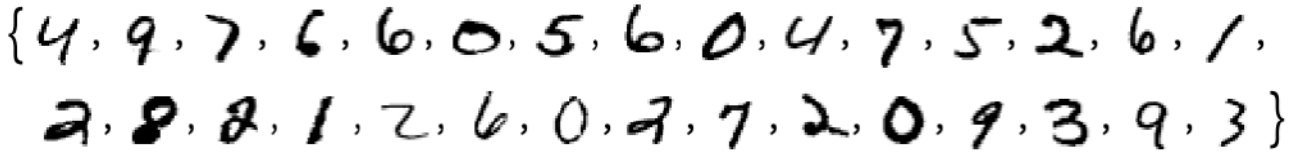
类人的任务模型

我们上面举的例子涉及到为数字数据建立模型，这些数据基本上来自于简单的物理学 —— 几个世纪以来我们都知道 “简单数学适用”。但是对于 ChatGPT 来说，我们必须为人类语言文本建立一个模型，即由人脑产生的那种模型。而对于这样的东西，我们（至少现在）还没有类似 “简单数学” 的东西。那么，它的模型可能是什么样的呢？

在我们谈论语言之前，让我们先谈谈另一项类似人类的任务：识别图像。而作为一个简单的例子，让我们考虑数字的图像（是的，这是一个经典的机器学习例子）：

我们可以做的一件事是为每个数字获取一堆样本图像：

然后，为了找出我们输入的图像是否对应于某个特定的数字，我们只需与我们拥有的样本进行明确的逐像素比较。但作为人类，我们似乎可以做得更好 —— 因为我们仍然可以识别数字，即使它们是手写的，并且有各种各样的修改和扭曲。



当我们为上面的数字数据建立一个模型时，我们能够取一个给定的数字值 x，然后为特定的 a 和 b 计算 a + bx。

因此，如果我们把这里的每个像素的灰度值当作某个变量 xi，是否有一些所有这些变量的函数，在评估时告诉我们这个图像是什么数字？事实证明，有可能构建这样一个函数。不足为奇的是，这并不特别简单。一个典型的例子可能涉及 50 万次数学运算。

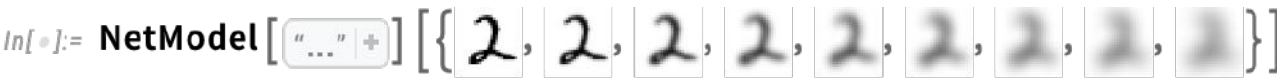
但最终的结果是，如果我们把一幅图像的像素值集合输入这个函数，就会得出一个数字，指定我们的图像是哪个数字。稍后，我们将讨论如何构建这样一个函数，以及神经网络的概念。但现在让我们把这个函数当作黑匣子，我们输入例如手写数字的图像（作为像素值的阵列），然后我们得到这些数字对应的数字：

```
In[•]:= NetModel[ "..." ][{ 7, 0, 9, 7, 8, 2, 4, 1, 1, 1 }]
```

```
Out[•]= {7, 0, 9, 7, 8, 2, 4, 1, 1, 1}
```

但这里到底发生了什么？比方说，我们逐步模糊一个数字。有一段时间，我们的函数仍然 “识别” 它，在这里是一个 “2”。但很快它就 “失去” 了，并开始给出 “错误” 的结果：

```
In[•]:= NetModel[ "..." ][{ 2, 2, 2, 2, 2, 2, 2, 2, 2 }]
```



```
Out[•]= {2, 2, 2, 1, 1, 1, 1, 1, 1}
```

但为什么我们说这是一个 “错误” 的结果呢？在这种情况下，我们知道我们通过模糊一个 “2” 得到所有的图像。但是，如果我们的目标是制作一个人类识别图像的模型，那么真正要问的问题是，如果遇到这些模糊的图像，在不知道其来源的情况下，人类会做什么。

如果我们从我们的功能中得到的结果通常与人类会说的话一致，我们就有一个 “好的模型”。而非微不足道的科学事实是，对于像这样的图像识别任务，我们现在基本上知道如何构建这样的函数。

我们能“从数学上证明”它们的作用吗？嗯，不能。因为要做到这一点，我们必须有一个关于我们人类正在做什么的数学理论。以“2”图像为例，改变几个像素。我们可以想象，只有几个像素“不合适”，我们还是应该认为这个图像是“2”。但这应该到什么程度呢？这是一个关于人类视觉感知的问题。而且，是的，对于蜜蜂或章鱼来说，答案无疑是不同的——对于假定的外星人来说，可能完全不同。

—4—

神经网路

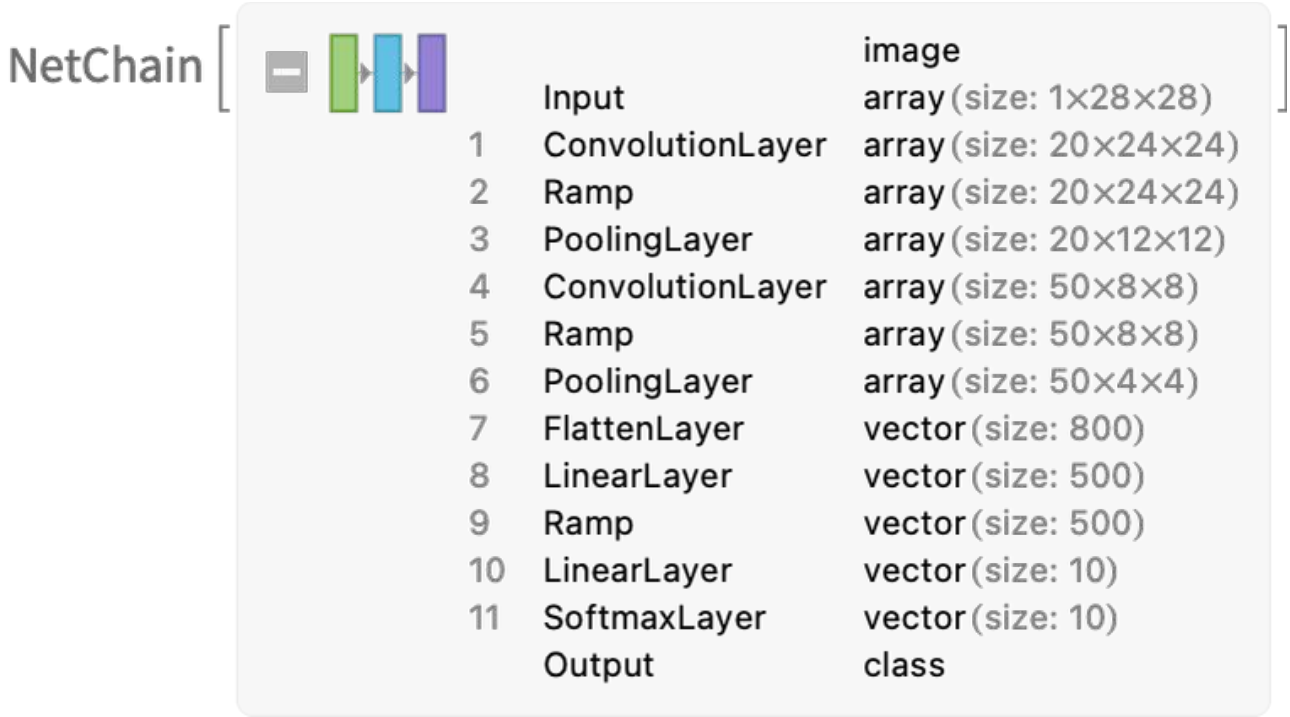
好吧，那么我们用于图像识别等任务的典型模型究竟是如何工作的呢？目前最流行、最成功的方法是使用神经网络。在 20 世纪 40 年代，神经网络的发明形式与今天的使用非常接近，它可以被认为是大脑似乎工作方式的简单理想化。

在人类的大脑中，有大约 1000 亿个神经元（神经细胞），每个神经元都能产生电脉冲，每秒可能有一千次。这些神经元在一个复杂的网络中连接起来，每个神经元都有树状的分支，允许它将电信号传递给可能有成千上万的其他神经元。

粗略估计，任何给定的神经元是否在某一时刻产生电脉冲，取决于它从其他神经元那里收到的脉冲 —— 不同的连接有不同的 “权重” 贡献。

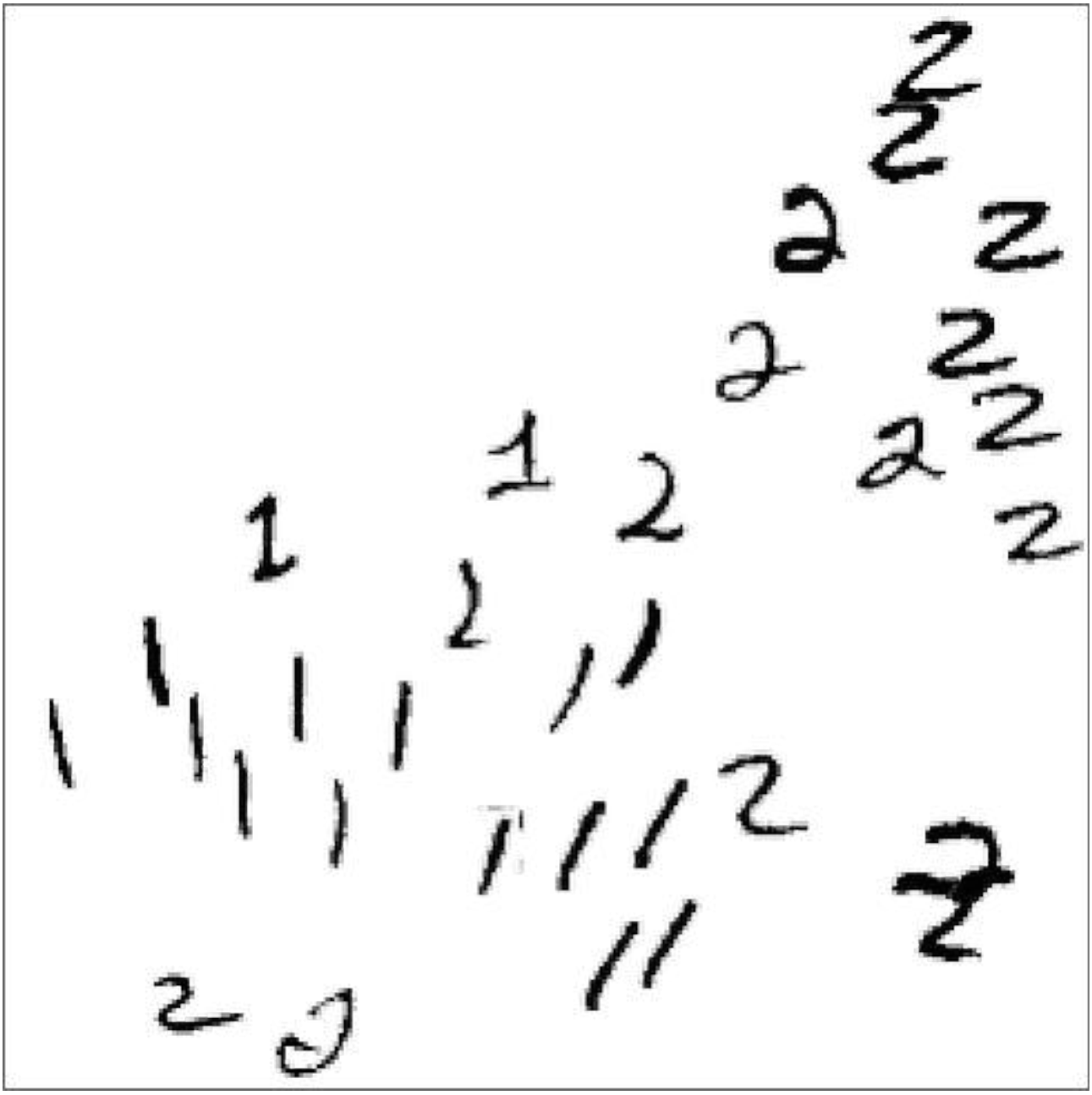
当我们 “看到一个图像” 时，所发生的事情是，当图像的光子落在眼睛后面的（“光感受器”）细胞上时，它们在神经细胞中产生电信号。这些神经细胞与其他神经细胞相连，最终信号通过一整层的神经元。而正是在这个过程中，我们 “识别” 了图像，最终 “形成了一个想法”，即我们 “看到了一个 2”（也许最后会做一些事情，如大声说 “2” 这个词）。

上一节中的 “黑盒子” 函数是这样一个神经网络的 “数学化” 版本。它刚好有 11 层（虽然只有 4 个 “核心层”）。



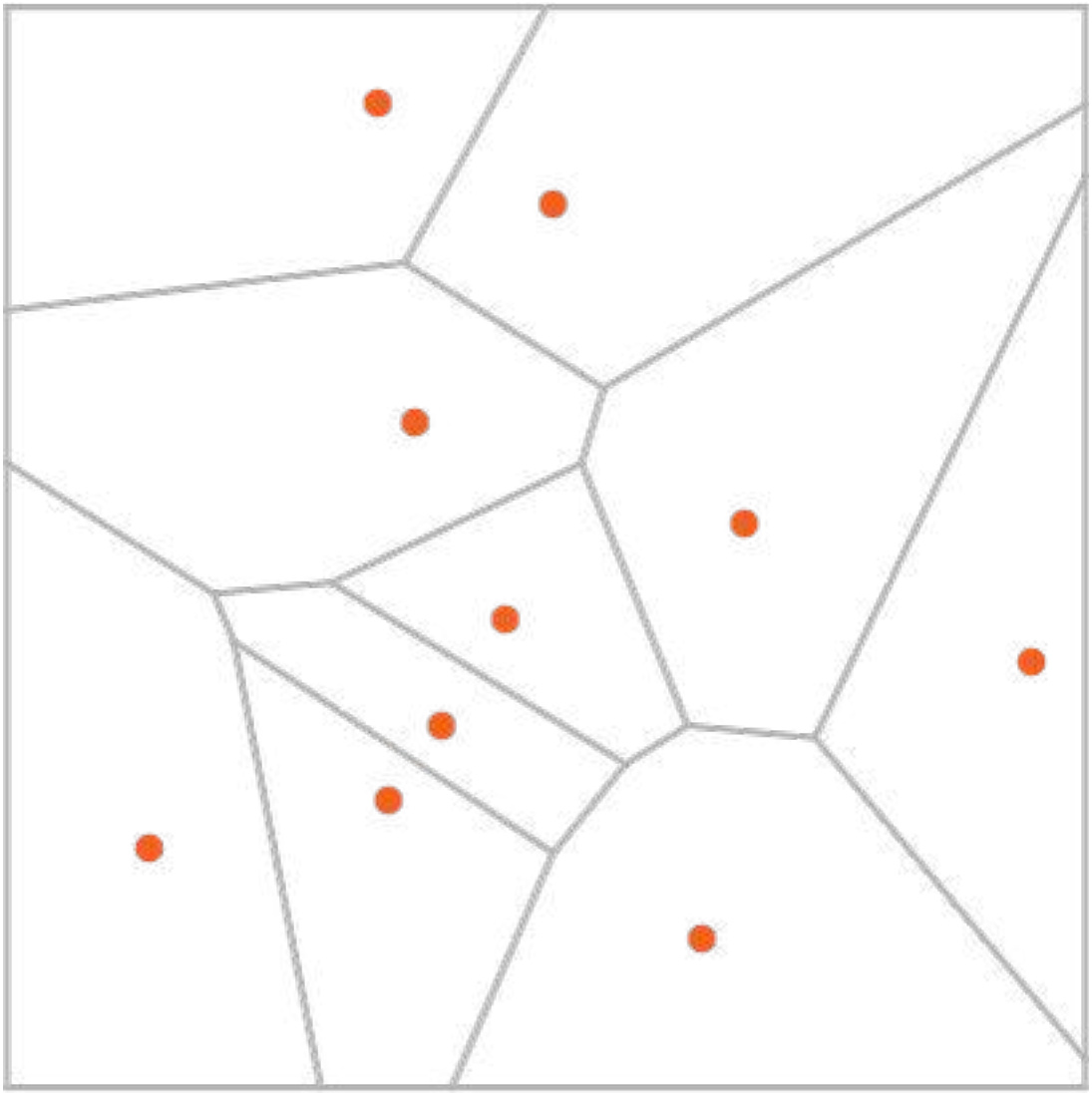
这个神经网并没有什么特别的 “理论推导”；它只是在 1998 年作为一项工程而构建的东西，并且被发现是有效的。（当然，这与我们描述我们的大脑是通过生物进化过程产生的没有什么不同）。

好吧，但是像这样的神经网络是如何“识别事物”的？关键在于吸引器的概念。想象一下，我们有 1 和 2 的手写图像：



我们希望所有的 1 都“被吸引到一个地方”，而所有的 2 都“被吸引到另一个地方”。或者，换一种方式，如果一个图像在某种程度上“更接近于 1”，而不是 2，我们希望它最终出现在“1 的地方”，反之亦然。

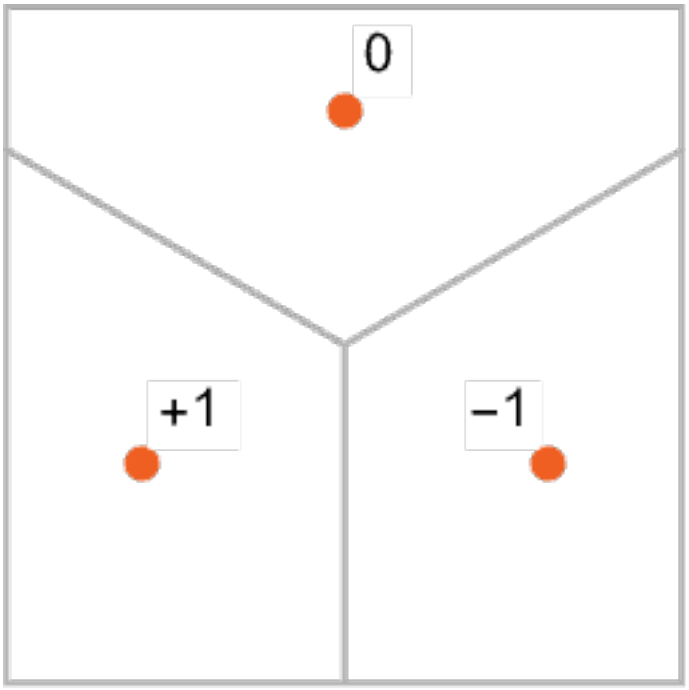
作为一个直接的类比，我们假设在平面上有某些位置，用点表示（在现实生活中，它们可能是咖啡店的位置）。那么我们可以想象，从平面上的任何一点开始，我们总是想在最近的点结束（即我们总是去最近的咖啡店）。我们可以通过将平面划分为由理想化的 “分水岭” 分隔的区域（“吸引盆地”）来表示这一点：



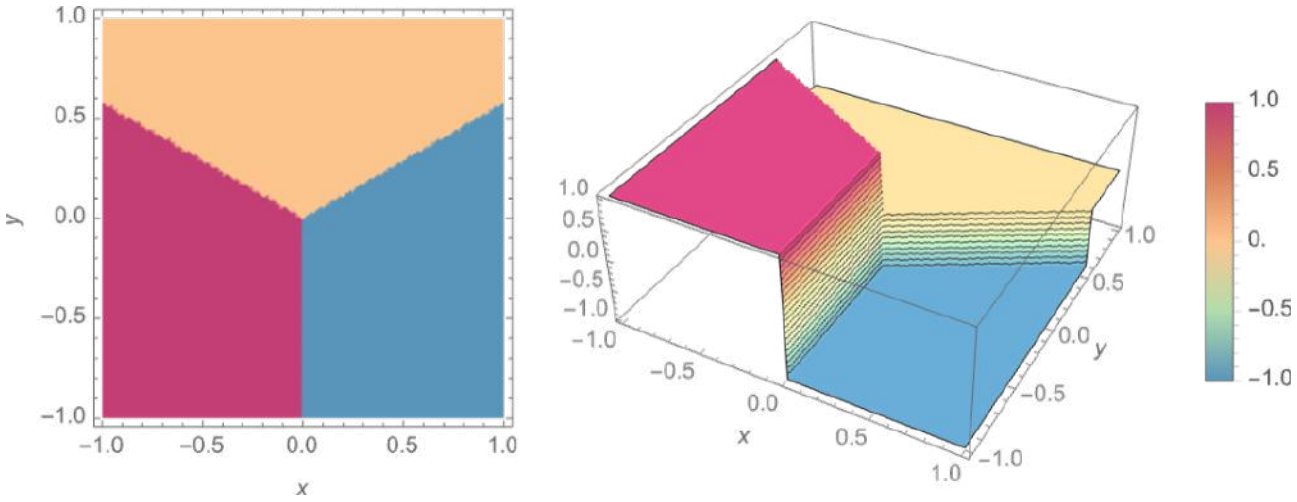
我们可以认为这是在执行一种 “识别任务”，我们不是在做类似于识别给定图像 “看起来最像” 的数字的事情 —— 而是很直接地看到给定点最接近哪个点。（我们在这里展示的 “Voronoi 图” 设置是在二维欧几里得空间中

分离点；数字识别任务可以被认为是在做非常类似的事情 —— 但却是在一个由每张图像中所有像素的灰度等级形成的 784 维空间中。）

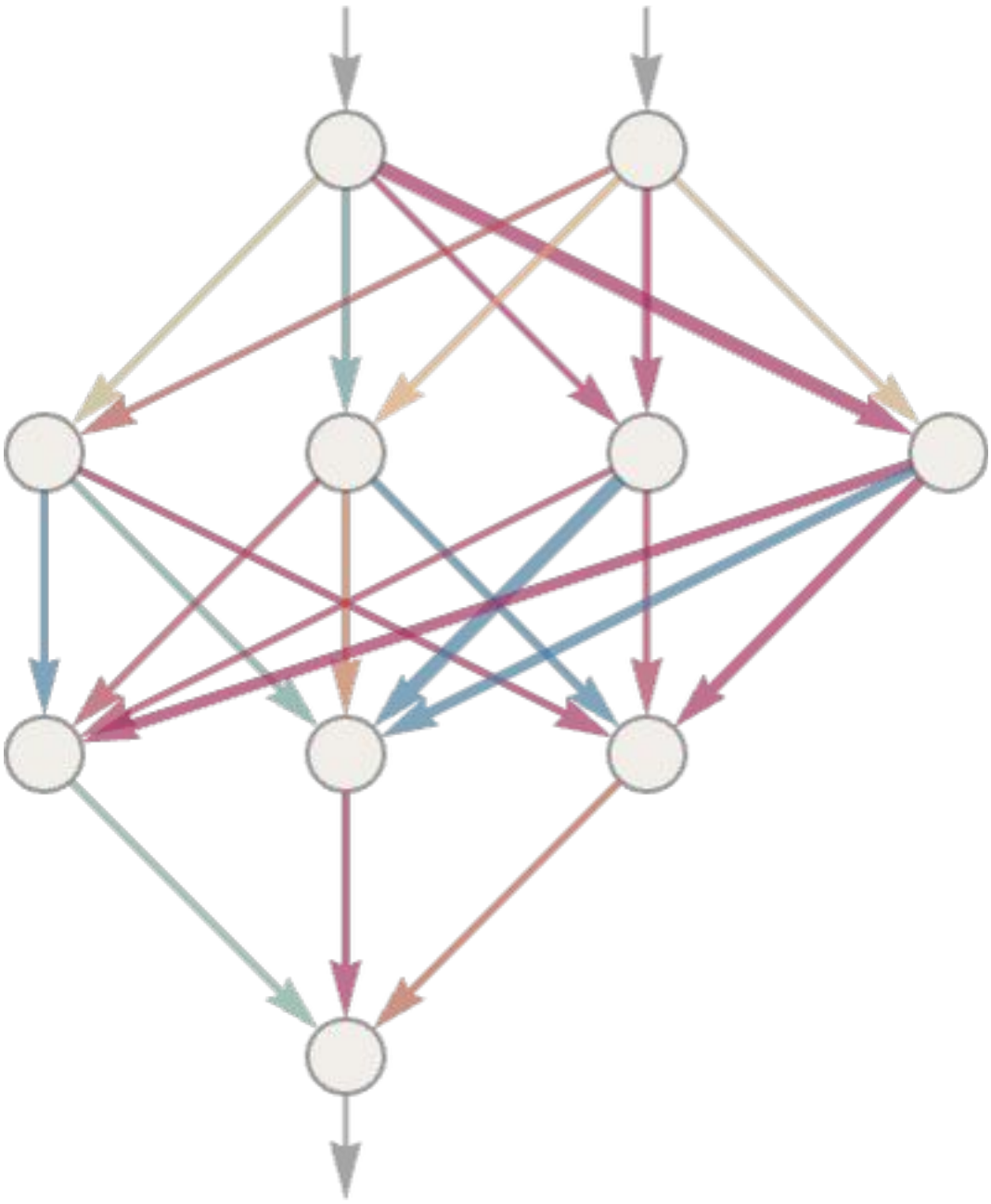
那么，我们如何使一个神经网络 “完成一个识别任务”？让我们考虑这个非常简单的案例：



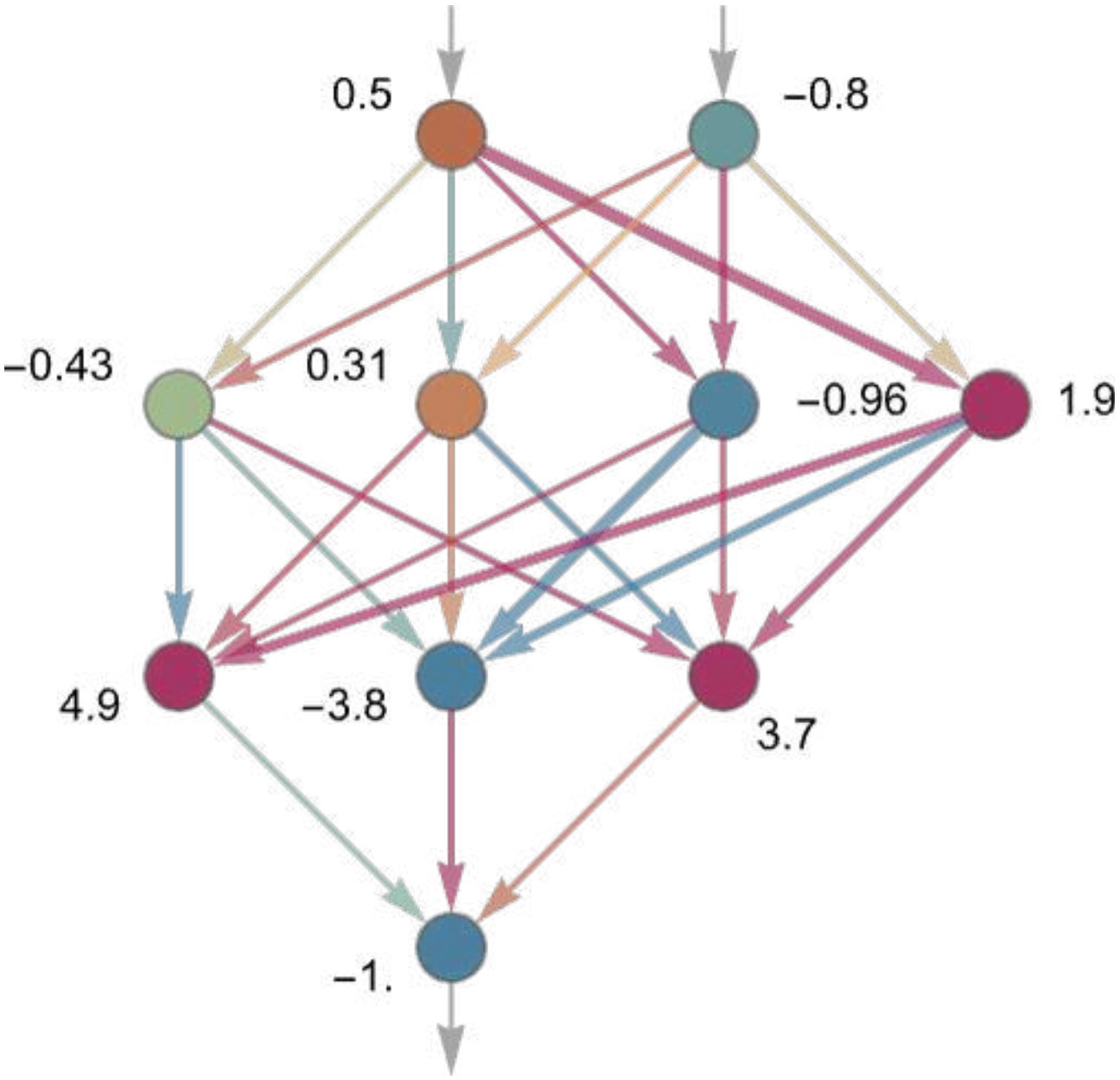
我们的目标是获取一个对应于 {x,y} 位置的 “输入”，然后将其 “识别” 为它最接近的三个点中的任何一个。或者，换句话说，我们希望神经网络能够计算出一个类似于 {x,y} 的函数：



那么，我们如何用神经网络做到这一点呢？归根结底，神经网是一个理想化的“神经元”的连接集合 —— 通常按层排列 —— 一个简单的例子是：



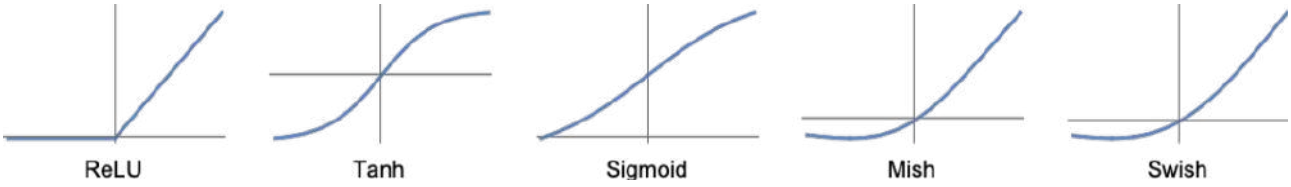
每个“神经元”都被有效地设置为评估一个简单的数字函数。为了“使用”这个网络，我们只需在顶部输入数字（如我们的坐标 x 和 y），然后让每一层的神经元“评估它们的功能”，并通过网络向前输入结果 —— 最终在底部产生最终的结果。



在传统的（受生物启发的）设置中，每个神经元实际上都有一组来自上一层神经元的“传入连接”，每个连接都被赋予一定的“权重”（可以是一个正数或负数）。一个给定的神经元的值是通过将“前一个神经元”的值乘以其相应的权重来确定的，然后将这些值相加并乘以一个常数，最后应用一个“阈值”（或“激活”）函数。

在数学术语中，如果一个神经元有输入 x = {x1, x2 …… }，那么我们计算 f[w.x + b]，其中权重 w 和常数 b 通常为网络中的每个神经元选择不同；函数 f 通常是相同的。

计算 w.x + b 只是一个矩阵乘法和加法的问题。激活函数 “f 引入了非线性（并最终导致了非线性行为）。通常使用各种激活函数；这里我们只使用 Ramp（或 ReLU）：



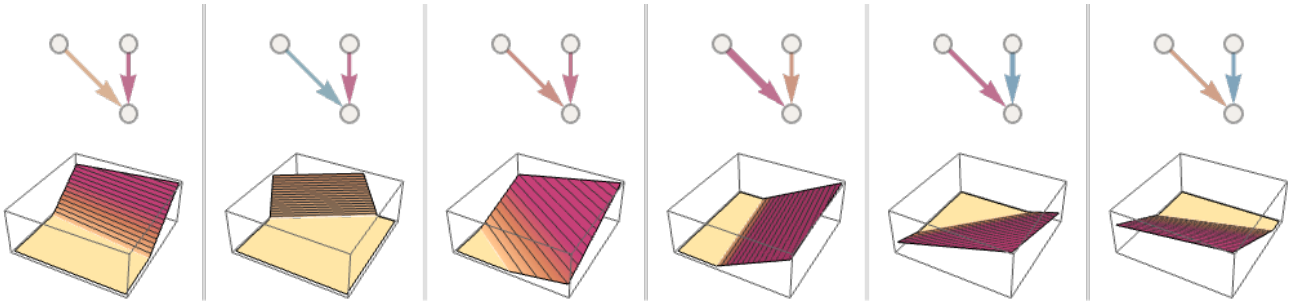
对于我们希望神经网络执行的每一项任务（或者说，对于我们希望它评估的每一个整体函数），我们将有不同的权重选择。（正如我们稍后要讨论的那样，这些权重通常是通过使用机器学习从我们想要的输出实例中 “训练” 神经网络来确定的）。

最终，每个神经网络都对应于一些整体的数学函数 —— 尽管它可能写得很乱。对于上面的例子，它就是：

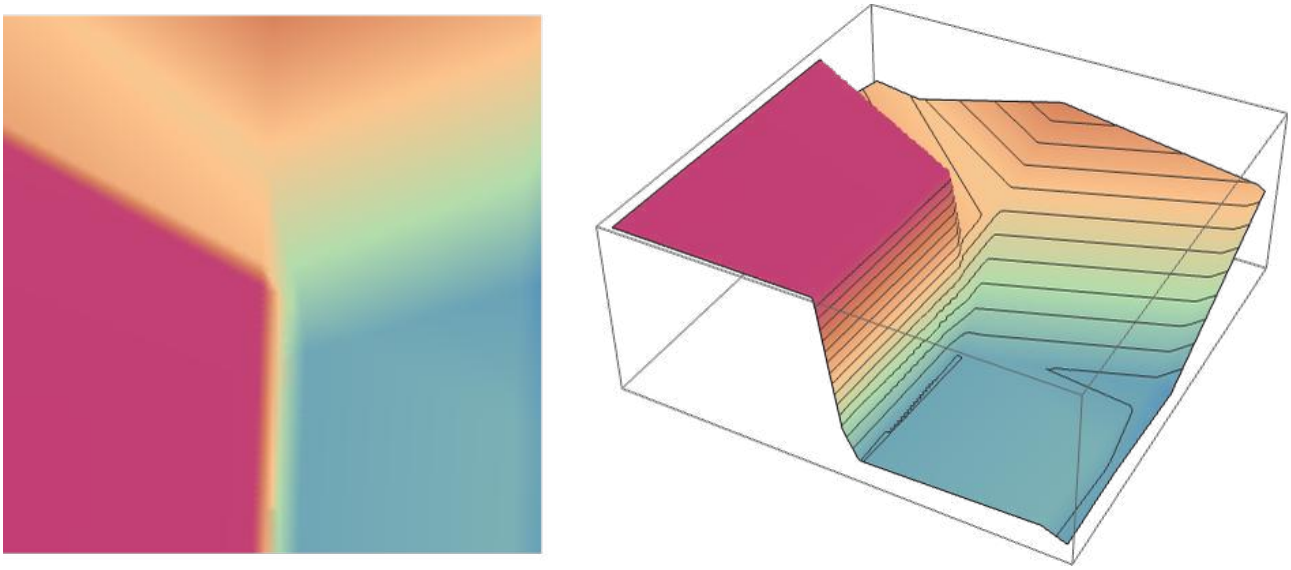
$$
\begin{aligned}
&w_{511} f(w_{311} f(b_{11} + x w_{111} + y w_{112}) + w_{312} f(b_{12} + x w_{121} + y w_{122}) + \\
&\qquad w_{313} f(b_{13} + x w_{131} + y w_{132}) + w_{314} f(b_{14} + x w_{141} + y w_{142}) + b_{31}) + \\
&w_{512} f(w_{321} f(b_{11} + x w_{111} + y w_{112}) + w_{322} f(b_{12} + x w_{121} + y w_{122}) + \\
&\qquad w_{323} f(b_{13} + x w_{131} + y w_{132}) + w_{324} f(b_{14} + x w_{141} + y w_{142}) + b_{32}) + \\
&w_{513} f(w_{331} f(b_{11} + x w_{111} + y w_{112}) + w_{332} f(b_{12} + x w_{121} + y w_{122}) + \\
&\qquad w_{333} f(b_{13} + x w_{131} + y w_{132}) + w_{334} f(b_{14} + x w_{141} + y w_{142}) + b_{33}) + b_{51}
\end{aligned}
$$

ChatGPT 的神经网络也只是对应于这样的一个数学函数 —— 但实际上有

但让我们回到单个神经元上。下面是一个有两个输入（代表坐标 x 和 y）的神经元在选择不同的权重和常数（以及 Ramp 作为激活函数）后可以计算的函数的一些例子：

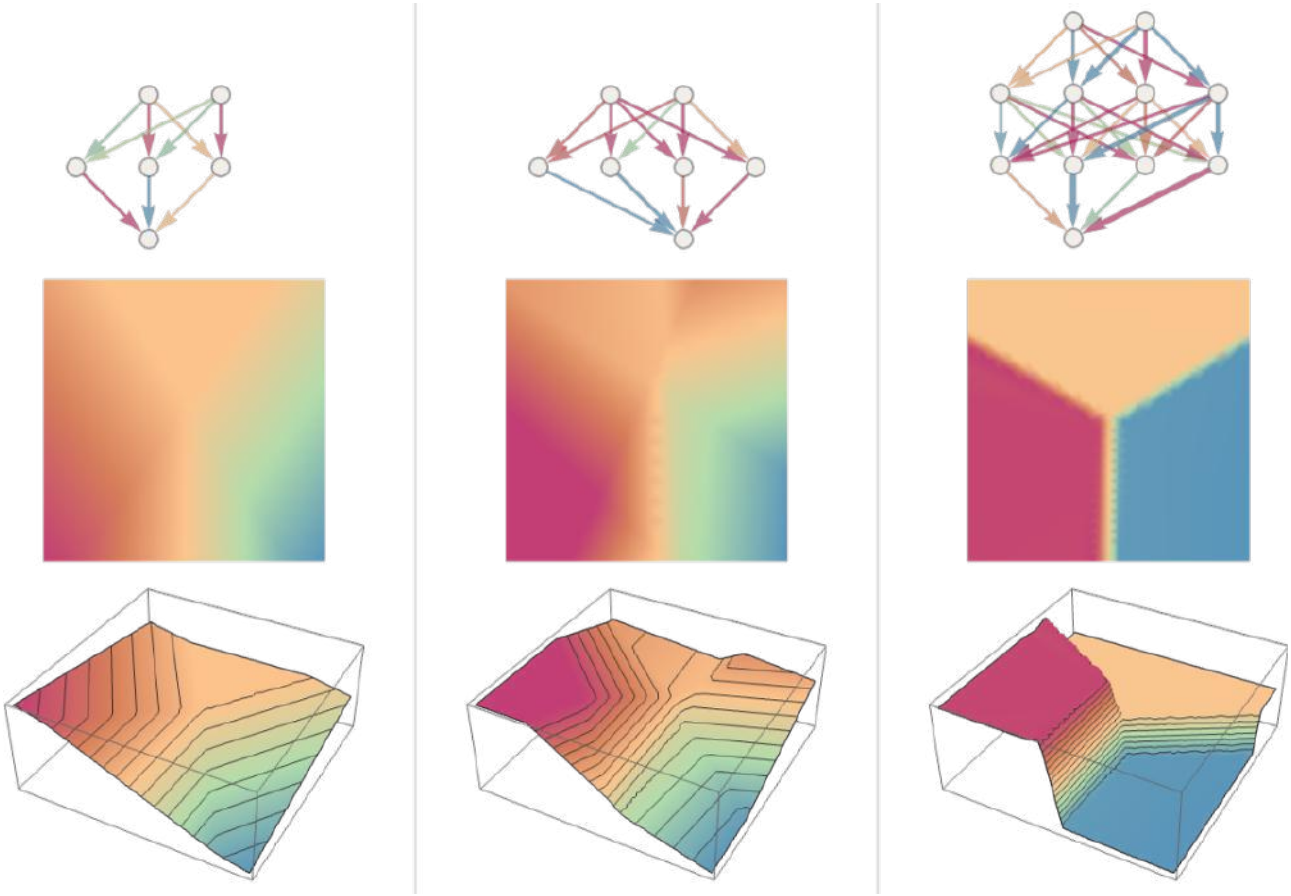


但是，上面那个更大的网络是怎么回事？嗯，这是它的计算结果：



这不是很“正确”，但它接近于我们上面展示的“最近点”函数。

让我们看看其他一些神经网络的情况。在每一种情况下，正如我们稍后所解释的，我们都在使用机器学习来寻找最佳的权重选择。然后，我们在这里展示带有这些权重的神经网络的计算结果：



更大的网络通常能更好地逼近我们的目标函数。而在“每个吸引子盆地的中间”，我们通常会得到我们想要的答案。但在边界 —— 神经网络“很难下定决心”的地方 —— 情况可能会更加混乱。

在这个简单的数学风格的“识别任务”中，“正确答案”是什么很清楚。但在识别手写数字的问题上，就不那么清楚了。如果有人把“2”写得很糟糕，看起来像“7”，等等，怎么办？不过，我们还是可以问，神经网络是如何区分数字的 —— 这就给出了一个指示：

我们能“从数学上”说说网络是如何区分的吗？并非如此。它只是在“做神经网络所做的事”而已。但事实证明，这通常似乎与我们人类所作的区分相当吻合。

让我们举一个更复杂的例子。比方说，我们有猫和狗的图像。我们有一个神经网络，它被训练来区分它们。下面是它在一些例子中可能做的事情：

现在，“正确答案” 是什么就更不清楚了。穿着猫衣的狗怎么办？等等。无论给它什么输入，神经网络都会产生一个答案。而且，事实证明，这样做的方式与人类可能做的事情是合理一致的。

正如我在上面所说的，这不是一个我们可以“从第一原理推导”的事实。它只是根据经验被发现是真的，至少在某些领域是这样。但这是神经网络有用的一个关键原因：它们以某种方式捕捉了“类似人类”的做事方式。

给自己看一张猫的照片，然后问 “为什么那是一只猫？”。也许你会开始说 “嗯，我看到它的尖耳朵，等等”。但要解释你是如何认出这张图片是一只猫的，并不是很容易。只是你的大脑不知怎么想出来的。但是对于大脑来说，没有办法（至少现在还没有）“进入” 它的内部，看看它是如何想出来的。

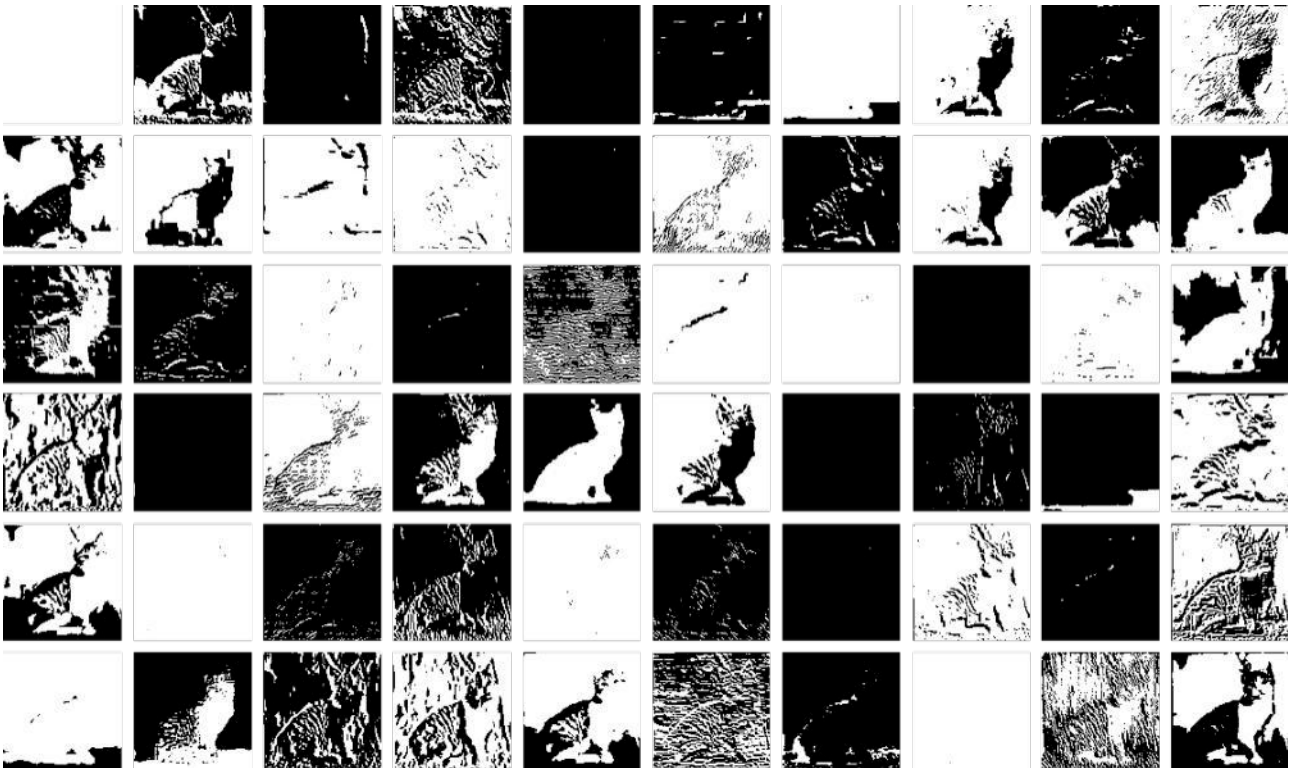
那么对于一个（人工）神经网来说呢？好吧，当你展示一张猫的图片时，可以直接看到每个 “神经元” 的作用。但是，即使要获得一个基本的可视化，通常也是非常困难的。

在我们用于解决上述 “最近点” 问题的最终网络中，有 17 个神经元。在用于识别手写数字的网络中，有 2190 个。而在我们用来识别猫和狗的网络中，有 60,650 个。

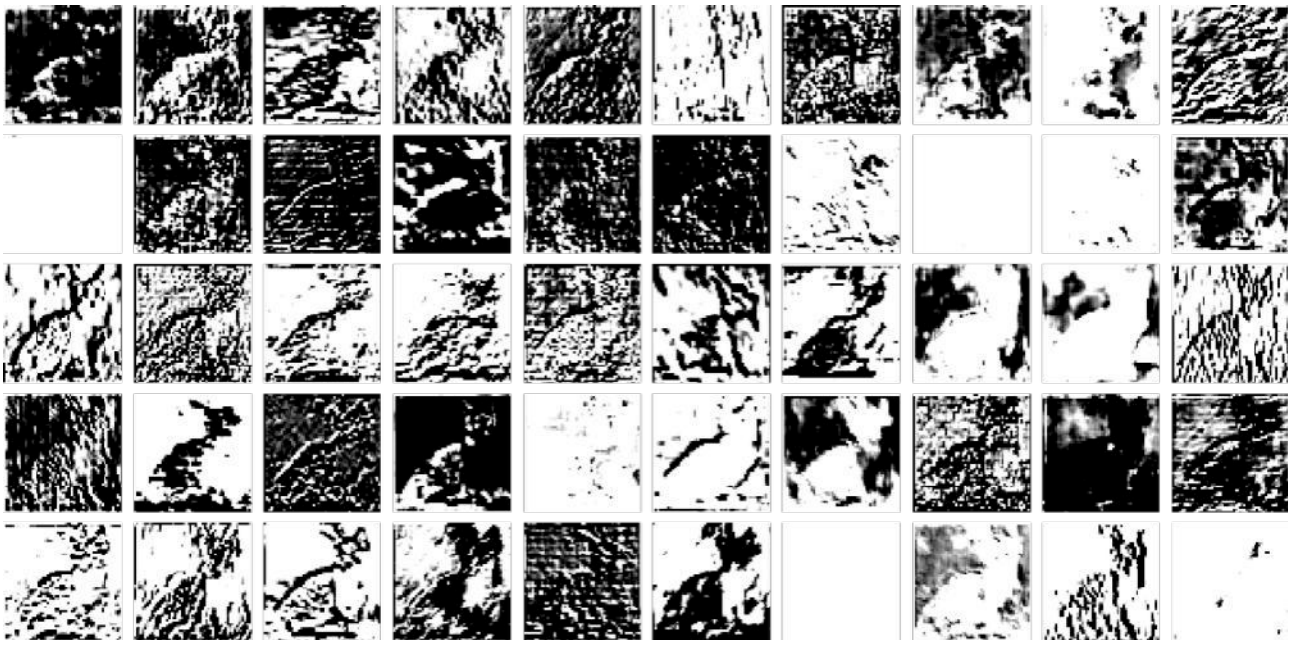
通常情况下，要将相当于 60,650 个维度的空间可视化是相当困难的。但由于这是一个为处理图像而设置的网络，它的许多神经元层被组织成阵列，就像它所看的像素阵列一样。

如果我们采取一个典型的猫图像：

那么我们就可以用一组衍生图像来表示第一层神经元的状态 —— 其中许多图像我们可以很容易地解释为“没有背景的猫”或“猫的轮廓”等：



到了第十层，就更难解释发生了什么：



但总的来说，我们可以说神经网络正在“挑选出某些特征”（也许尖尖的耳朵也在其中），并利用这些特征来确定图像是什么。但这些特征是我们有名字的，比如“尖耳朵”？大多数情况下不是。

我们的大脑在使用类似的特征吗？大多数情况下我们不知道。但值得注意的是，像我们在这里展示的神经网络的前几层似乎可以挑出图像的某些方面（如物体的边缘），这些方面似乎与我们知道的由大脑中第一层视觉处理挑出的特征相似。

但是，假设我们想要一个神经网络的“猫识别理论”。我们可以说“看，这个特定的网络做到了”——这立即给了我们一些关于“问题有多难”的感觉（例如，可能需要多少个神经元或层）。

但至少到现在为止，我们还没有办法对网络正在做的事情进行“叙述性描述”。也许这是因为它在计算上确实是不可简化的，而且除了明确地追踪每一个步骤之外，没有一般的方法可以找到它在做什么。也可能只是因为我们还没有“弄清科学”，还没有确定“自然法则”，使我们能够总结出正在发生的事情。

当我们谈论用 ChatGPT 生成语言时，我们会遇到同样的问题。而且同样不清楚是否有办法“总结它在做什么”。但是语言的丰富性和细节（以及我们在这方面的经验）可能会让我们比图像走得更远。

— 5 —

机器学习和神经网络的训练

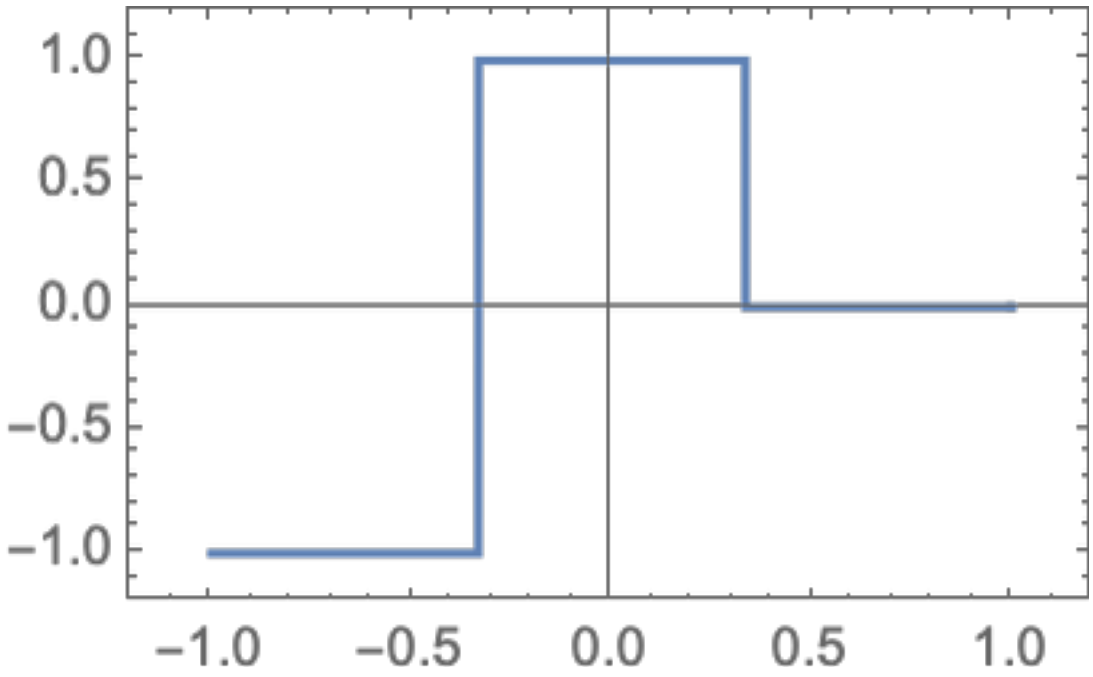
到目前为止，我们一直在谈论那些“已经知道”如何完成特定任务的神经网络。但是，神经网络之所以如此有用（估计也是在大脑中），是因为它们不仅在原则上可以完成各种任务，而且可以逐步“根据实例训练”来完成这些任务。

当我们制作一个区分猫和狗的神经网络时，我们实际上不需要写一个程序来（比如说）明确地找到胡须；相反，我们只需要展示大量关于什么是猫和什么是狗的例子，然后让网络从这些例子中“机器学习”如何去区分它们。

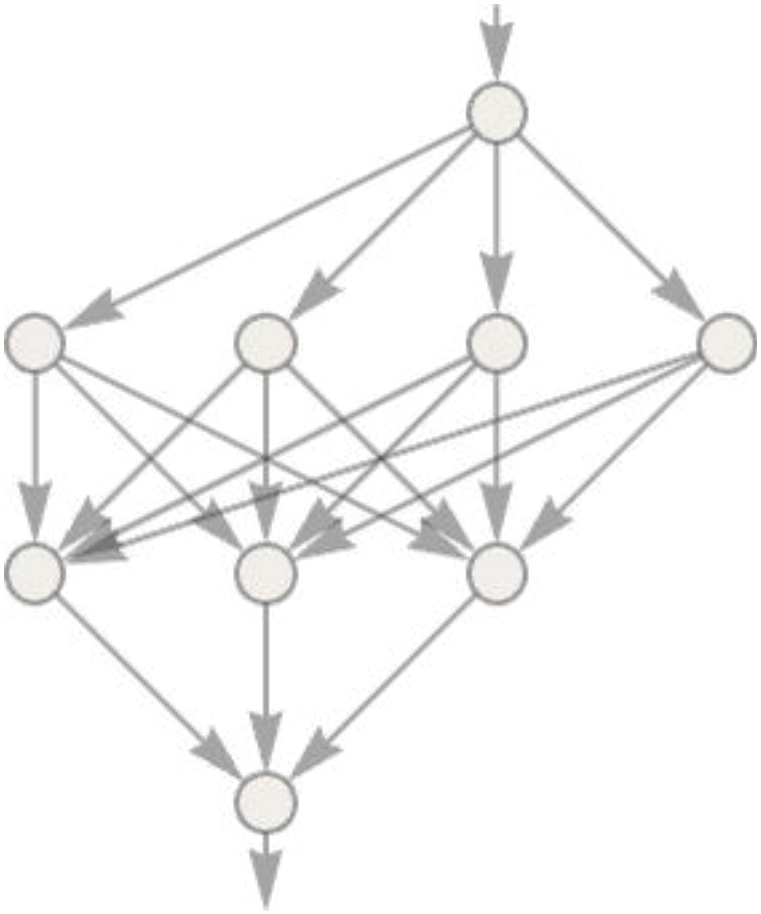
重点是，训练有素的网络从它所展示的特定例子中“概括”出来。正如我们在上面看到的，这并不是简单地让网络识别它所看到的猫咪图像的特定像素模式；而是让神经网络以某种方式设法在我们认为是某种“一般猫性”的基础上区分图像。

那么，神经网络的训练究竟是如何进行的呢？从本质上讲，我们一直在努力寻找能够使神经网络成功重现我们所给的例子的权重。然后，我们依靠神经网络以“合理”的方式在这些例子之间进行“插值”（或“概括”）。

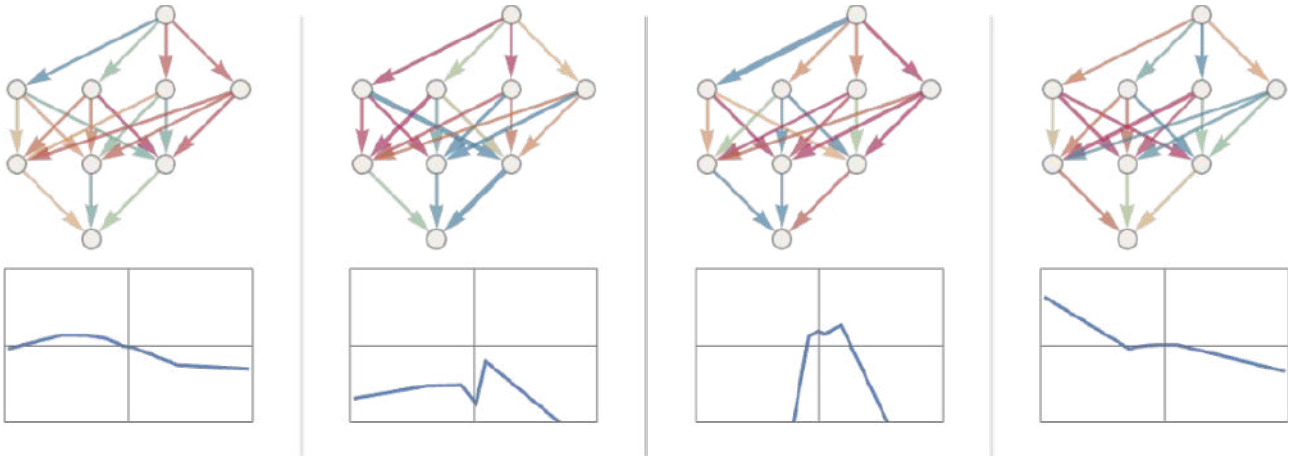
让我们看看一个比上面的最近点的问题更简单的问题。让我们只尝试让一个神经网络学习函数：



对于这个任务，我们需要一个只有一个输入和一个输出的网络，比如：

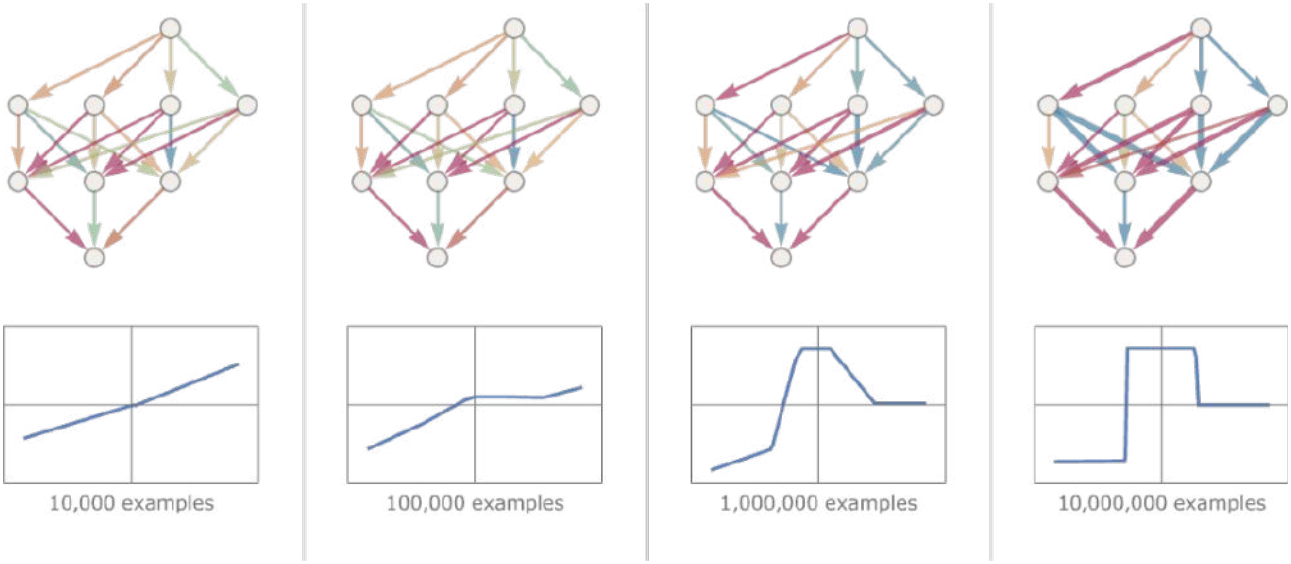


但我们应该使用什么权重等？在每一组可能的权重下，神经网络都会计算出一些函数。例如，这里是它用几组随机选择的权重所做的事情：



是的，我们可以清楚地看到，在这些情况下，它甚至都没有接近再现我们想要的函数。那么，我们如何找到能够重现该功能的权重呢？

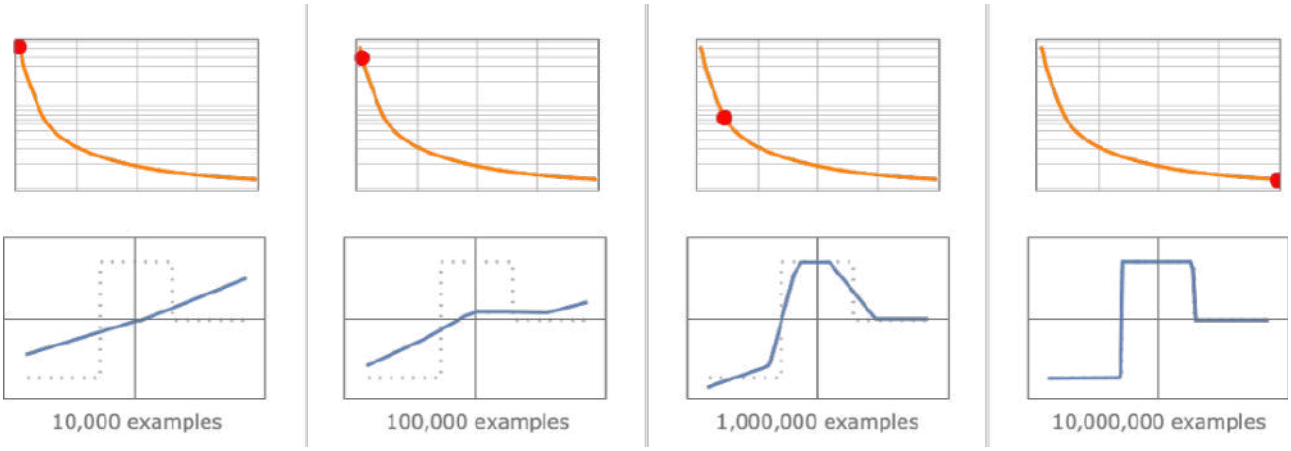
基本的想法是提供大量的“输入→输出”的例子来“学习”——然后尝试找到能重现这些例子的权重。下面是用逐渐增多的例子来做的结果：



在这个“训练”的每个阶段，网络中的权重都被逐步调整——我们看到，最终我们得到了一个能成功重现我们想要的功能的网络。那么，我们是如何调整权重的呢？基本的想法是在每个阶段看看我们离得到我们想要的功能“有多远”，然后以这样的方式更新权重，使之更接近。

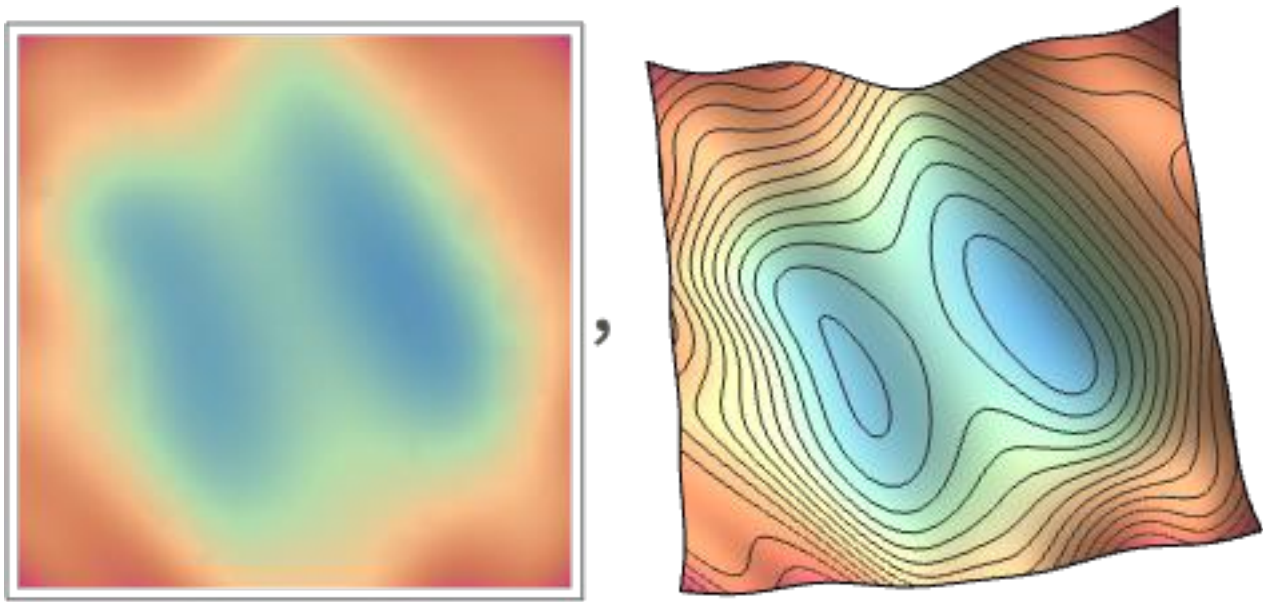
为了找出“我们有多远”，我们计算通常被称为“损失函数”（或有时称为“成本函数”）的东西。这里我们使用的是一个简单的（L2）损失函数，它只是我们得到的值与真实值之间的差异的平方之和。

我们看到的是，随着我们训练过程的进展，损失函数逐渐减少（遵循一定的“学习曲线”，不同的任务是不同的）——直到我们达到一个点，网络（至少是一个很好的近似值）成功再现了我们想要的函数：

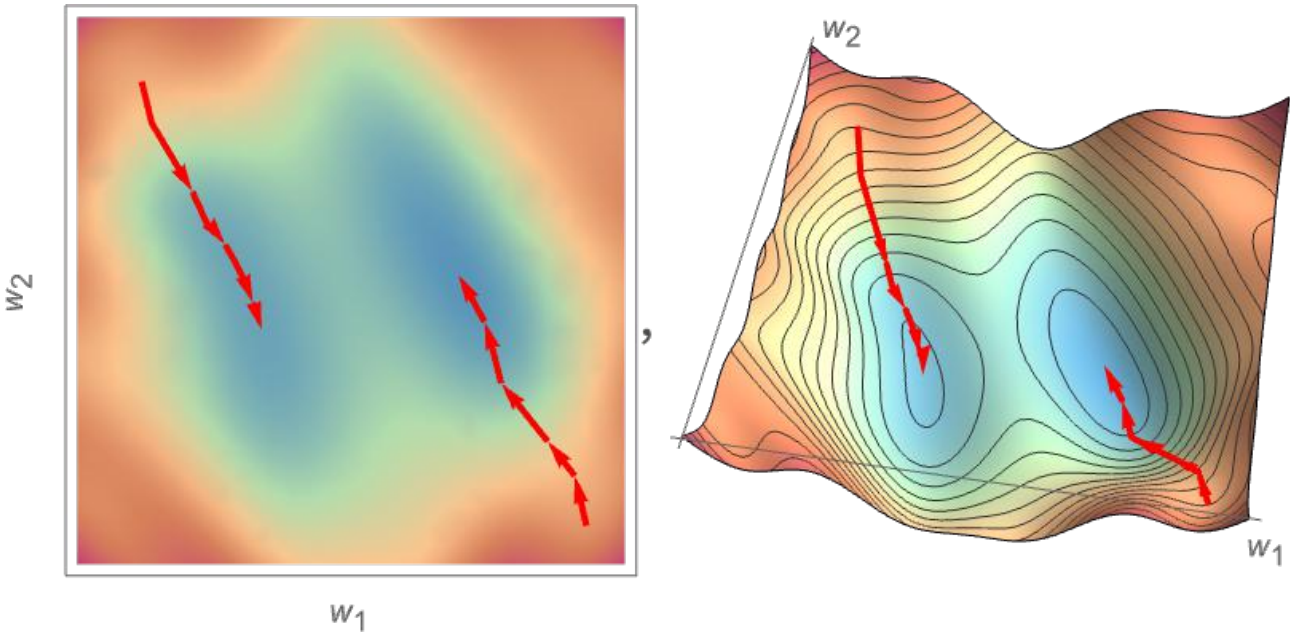


好了，最后要解释的重要部分是如何调整权重以减少损失函数。正如我们所说，损失函数给我们提供了我们得到的值与真实值之间的“距离”。但是“我们得到的值”在每个阶段都是由当前版本的神经网络和其中的权重决定的。但现在想象一下，这些权重是变量——比如说 wi。我们想找出如何调整这些变量的值，以使取决于这些变量的损失最小。

例如，想象一下（对实践中使用的典型神经网络进行了不可思议的简化），我们只有两个权重 w1 和 w2。那么我们可能有一个损失，作为 w1 和 w2 的函数，看起来像这样：



数值分析提供了各种技术来寻找这样的情况下的最小值。但一个典型的方法是，从之前的 w1、w2 开始，逐步遵循最陡峭的下降路径：



就像水从山上流下来一样，所能保证的是这个过程最终会在地表的某个局部最小值（“一个山湖”）；它很可能达不到最终的全球最小值。

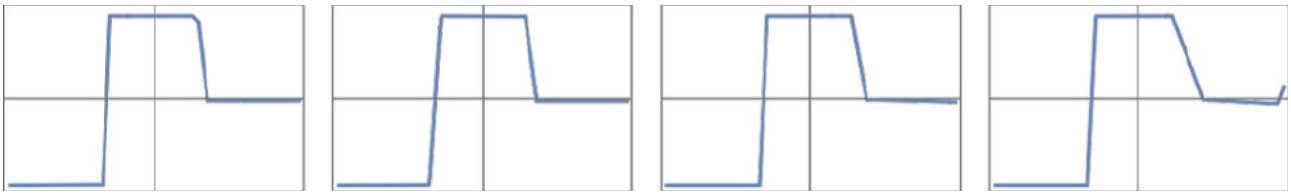
在 “重量景观” 上找到最陡峭的下降路径并不明显，这是不可行的。但是，微积分可以帮助我们。正如我们上面提到的，我们总是可以把神经网看作是在计算一个数学函数 —— 它取决于它的输入和权重。

但现在考虑对这些权重进行微分。事实证明，微积分的连锁法则实际上可以让我们 “解开” 神经网中连续几层所做的运算。其结果是，我们可以 —— 至少在某些局部近似中 —— “反转” 神经网的操作，并逐步找到使与输出相关的损失最小的权重。

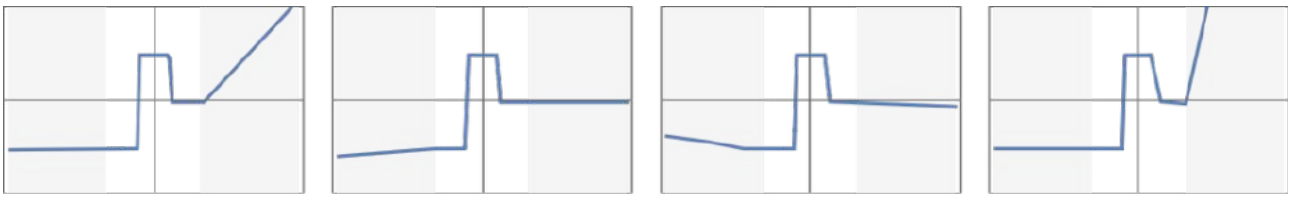
上图显示了在只有 2 个权重的不现实的简单情况下，我们可能需要做的最小化工作。但事实证明，即使有更多的权重（ChatGPT 使用了 1750 亿个），仍有可能做到最小化，至少在某种程度上是近似的。事实上，2011 年左右发生的 “深度学习” 的重大突破与以下发现有关：从某种意义上说，当有很多权重参与时，做（至少是近似）最小化比有相当少的权重更容易。

换句话说 —— 有点反直觉 —— 用神经网络解决更复杂的问题比简单的问题更容易。其大致原因似乎是，当一个人有很多 “权重变量” 时，他有一个高维空间，有 “很多不同的方向”，可以把他引向最小值 —— 而如果变量较少，则更容易陷入一个局部最小值（“山湖”），没有 “方向可以出去”。

值得指出的是，在典型的情况下，有许多不同的权重集合，它们都能使神经网络具有几乎相同的性能。而在实际的神经网络训练中，通常会有很多随机的选择，导致 “不同但等同的解决方案”，就像这些：



但每一个这样的 “不同的解决方案” 至少会有轻微的不同行为。如果我们要求，比如说，在我们提供训练实例的区域之外进行 “外推”，我们可以得到极大的不同结果：



但是哪一个是“正确的”呢？真的没有办法说。它们都“与观察到的数据一致”。但它们都对应着不同的“先天”方式来“思考”如何在“盒子外”做什么。对我们人类来说，有些可能比其他的看起来“更合理”。

— 6 —

神经网络训练的实践与理论

特别是在过去的十年里，在训练神经网络的艺术方面取得了许多进展。而且，是的，这基本上是一门艺术。有时，特别是在回顾中，人们至少可以看到正在做的事情有一丝“科学解释”的影子。但大多数情况下，事情都是通过试验和错误发现的，增加了一些想法和技巧，逐步建立了一个关于如何使用神经网络的重要传说。

有几个关键部分。首先，对于一个特定的任务，应该使用什么架构的神经网络。然后，还有一个关键问题，即如何获得训练神经网络的数据。而且，人们越来越多地不是在处理从头开始训练一个网络的问题：相反，一

个新的网络可以直接纳入另一个已经训练好的网络，或者至少可以使用该网络为自己产生更多的训练实例。

人们可能认为，对于每一种特定的任务，人们都需要一个不同的神经网络结构。但人们发现，即使是对于明显不同的任务，相同的架构似乎也能发挥作用。

在某种程度上，这让人想起了通用计算的想法（以及我的计算等价原则），但是，正如我将在后面讨论的那样，我认为这更多地反映了这样一个事实，即我们通常试图让神经网络做的任务是“类似人类”的，而神经网络可以捕获相当普遍的“类似人类的过程”。

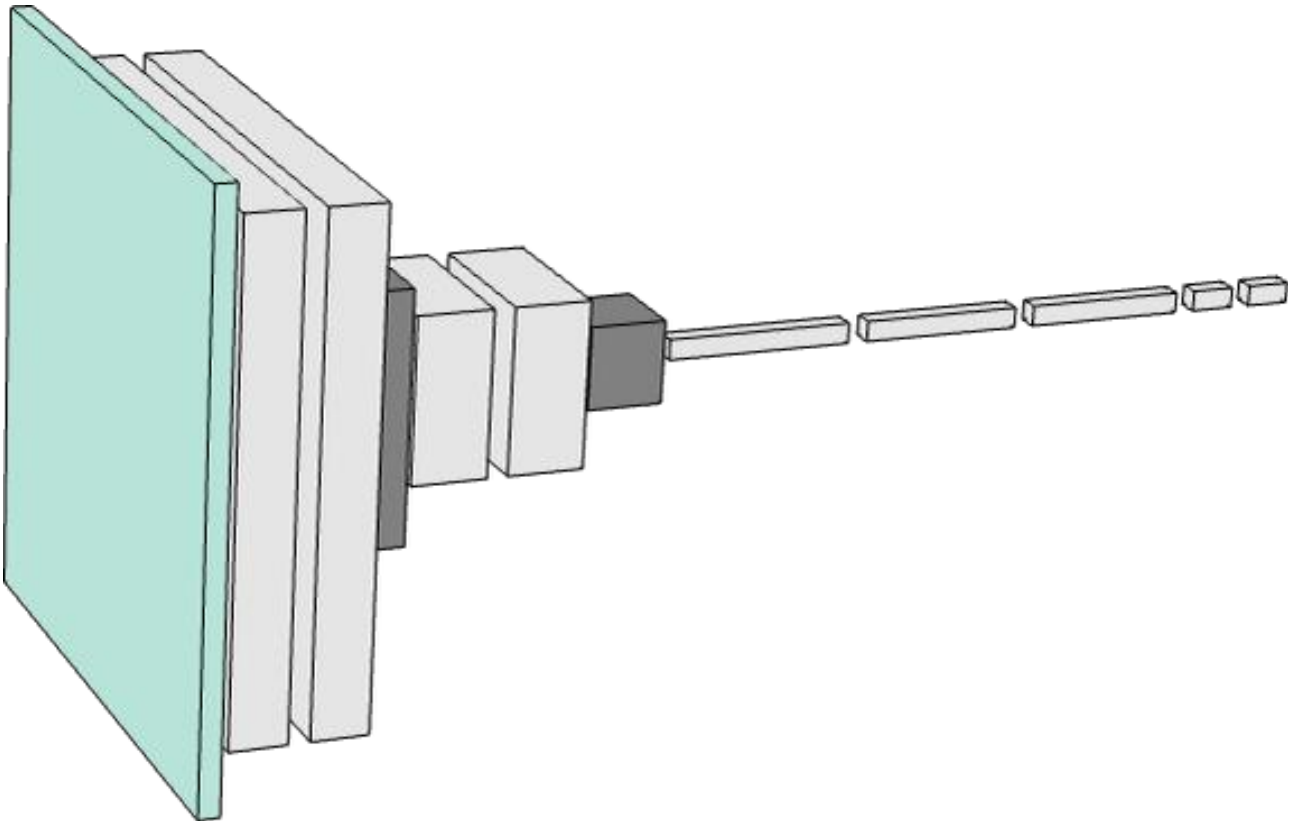
在早期的神经网络中，人们倾向于认为应该“让神经网络尽可能地少做”。例如，在将语音转换为文本时，人们认为应该首先分析语音的音频，将其分解为音素，等等。但人们发现，至少对于“类似人类的任务”来说，通常更好的做法是尝试在“端到端问题”上训练神经网络，让它自己“发现”必要的中间特征、编码等。

还有一个想法是，我们应该在神经网络中引入复杂的单独组件，让它实际上“明确地实现特定的算法想法”。但是，这又一次被证明是不值得的；相反，最好只是处理非常简单的组件，让它们“自我组织”（尽管通常是以我们无法理解的方式）来实现（大概）那些算法想法的等价物。

这并不是说没有与神经网络相关的“结构化思想”。因此，例如，具有局部连接的二维神经元阵列似乎至少在处理图像的早期阶段非常有用。而拥

有专注于“回顾序列”的连接模式似乎很有用——我们将在后面看到——在处理人类语言等事物时，例如在 ChatGPT 中。

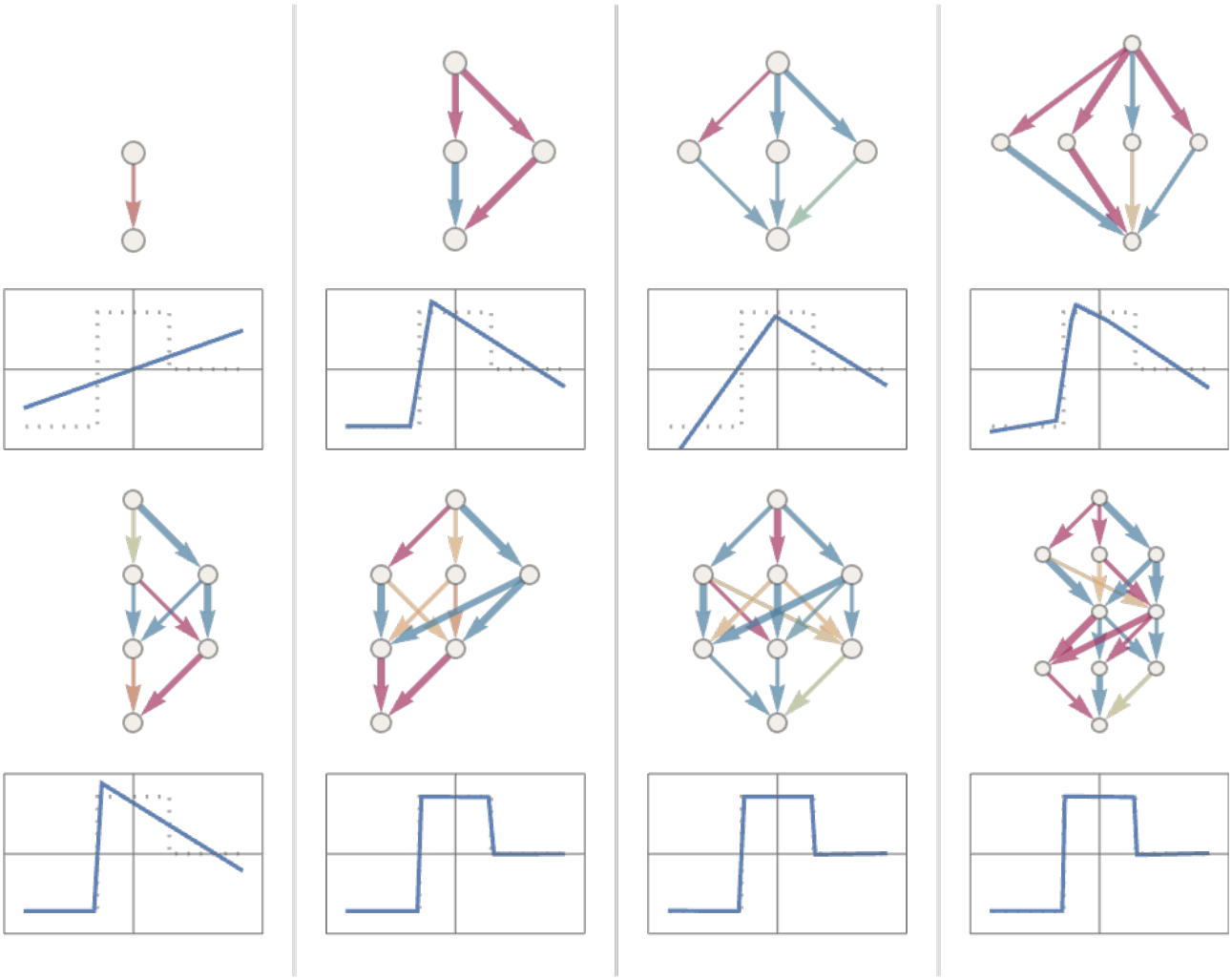
但神经网络的一个重要特点是，像一般的计算机一样，它们最终只是在处理数据。而目前的神经网络——目前的神经网络训练方法——是专门处理数字阵列的。但在处理过程中，这些数组可以被完全重新排列和重塑。举个例子，我们上面用来识别数字的网络从一个二维的“图像”阵列开始，迅速“增厚”到许多通道，但随后“浓缩”成一个一维阵列，最终将包含代表不同可能输出数字的元素：



但是，好吧，如何判断一个特定的任务需要多大的神经网？这是一门艺术。在某种程度上，关键是要知道“这个任务有多难”。但对于类似人类的任务来说，这通常是很难估计的。

是的，可能有一种系统的方法可以通过计算机非常“机械”地完成任务。但很难知道是否存在人们认为的技巧或捷径，使人们至少在“类似人类的水平”上更容易地完成这项任务。可能需要列举一个巨大的游戏树来“机械地”玩某个游戏；但可能有一个更容易（“启发式”）的方法来实现“人类水平的游戏”。

当人们在处理微小的神经网络和简单的任务时，有时可以明确地看到“从这里不能到达那里”。例如，这是人们在上一节的任务中用几个小的神经网络似乎能做到的最好的结果：



而我们的情况是，如果网太小，它就不能再现我们想要的功能。但如果超过一定的规模，它就没有问题了 —— 至少如果一个人用足够长的时间和足够多的例子训练它。顺便说一下，这些图片说明了一个神经网络的传说：如果中间有一个 “挤压”，迫使所有东西都通过一个较小的中间神经元数量，那么我们往往可以用一个较小的网络。

（值得一提的是，“无中间层” —— 或所谓的 “感知器” —— 网络只能学习本质上的线性函数 —— 但只要有一个中间层，原则上就可以任意很好地近似任何函数，至少如果有足够的神经元，尽管为了使其可行地训练，通常需要某种正则化或规范化）。

好吧，让我们假设我们已经确定了某种神经网络架构。现在有一个问题，就是如何获得数据来训练网络。围绕神经网络和一般机器学习的许多实际挑战都集中在获取或准备必要的训练数据上。在许多情况下（“监督学习”），人们希望获得明确的输入和期望的输出的例子。

因此，举例来说，人们可能希望通过图像中的内容或一些其他属性来标记图像。也许我们必须明确地去做 —— 通常是费尽心机地去做标记。但是很多时候，我们可以借助已经完成的工作，或者将其作为某种代理。

因此，举例来说，我们可以使用网络上已经提供的图片的 alt 标签。或者，在另一个领域，我们可以使用为视频创建的封闭式字幕。或者在语言翻译训练中，可以使用不同语言的网页或其他文件的平行版本。

你需要向神经网络展示多少数据来训练它完成一项特定任务？同样，这很难从第一原理上估计。当然，通过使用“转移学习”来“转移”诸如已经在另一个网络中学习过的重要特征列表的东西，可以大大降低要求。

但一般来说，神经网络需要“看到大量的例子”才能训练好。而至少对于某些任务来说，神经网络的一个重要传说是，这些例子可能是非常重复的。事实上，向神经网络展示所有的例子是一个标准的策略，一遍又一遍。在每个“训练回合”（或“epochs”）中，神经网络至少会处于一个稍微不同的状态，而以某种方式“提醒”它某个特定的例子对于让它“记住那个例子”是很有用的。（是的，也许这类似于人类记忆中的重复的有用性）。

但往往只是反复重复同一个例子是不够的。还需要向神经网络展示这个例子的变化。而神经网络理论的一个特点是，这些“数据增强”的变化不一定要复杂才有用。只要用基本的图像处理方法稍微修改一下图像，就可以使它们在神经网络训练中基本上“像新的一样好”。同样，当人们没有实际的视频等来训练自动驾驶汽车时，人们可以继续从模拟的视频游戏环境中获得数据，而不需要实际的真实世界场景的所有细节。

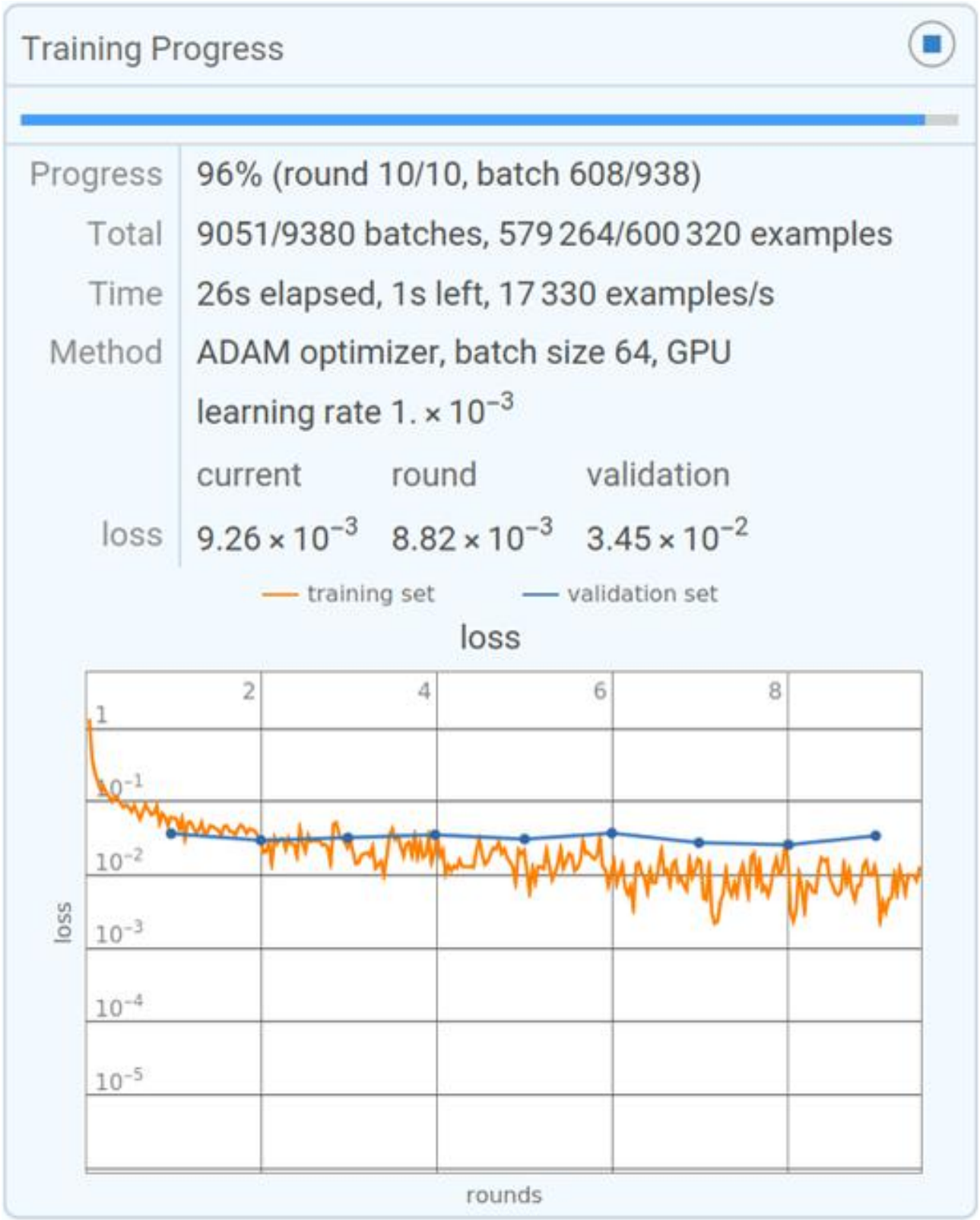
像 ChatGPT 这样的东西如何呢？嗯，它有一个很好的特点，那就是它可以进行“无监督学习”，这使得它更容易得到用于训练的例子。回顾一下，ChatGPT 的基本任务是找出如何继续它所给的一段文字。因此，为了获得“训练实例”，我们所要做的就是获得一段文本，并将其结尾遮盖起来，然后将其作为“训练的输入”——“输出”是完整的、未被遮盖的文本。我们稍后会详细讨论这个问题，但主要的一点是，与学习图片中的内容不同，

不需要“明确的标签”；ChatGPT 实际上可以直接从它所得到的任何文本例子中学习。

好吧，那么神经网络的实际学习过程是怎样的呢？归根结底，这都是为了确定什么权重能够最好地捕捉所给的训练实例。有各种详细的选择和“超参数设置”（之所以被称为超参数，是因为可以把权重看作是“参数”），可以用来调整如何完成这一过程。

有不同的损失函数选择（平方之和、绝对值之和，等等）。有不同的方法来进行损失最小化（每一步要在权重空间中移动多远，等等）。然后还有一些问题，比如要展示多大的“一批”例子来获得每一个试图最小化的损失的连续估计。而且，是的，人们可以应用机器学习（例如，我们在 Wolfram 语言中所做的）来实现机器学习的自动化 —— 自动设置超参数等东西。

但最终，整个训练过程的特点是看到损失是如何逐渐减少的（如这个 Wolfram Language 的小型训练的进度监视器）：



而人们通常看到的是，损失在一段时间内减少，但最终在某个恒定值上趋于平缓。如果这个值足够小，那么可以认为训练是成功的；否则，这可能是一个应该尝试改变网络结构的信号。

能否告诉我们“学习曲线”要花多长时间才能变平？就像许多其他事情一样，似乎有近似的幂律缩放关系，这取决于神经网络的大小和使用的数据量。但一般的结论是，训练一个神经网络是很难的，需要大量的计算努力。作为一个实际问题，这些努力的绝大部分都花在了对数字阵列的操作上，而这正是 GPU 所擅长的 —— 这就是为什么神经网络训练通常受限于 GPU 的可用性。

在未来，是否会有从根本上更好的方法来训练神经网络，或者一般地做神经网络的工作？我认为，几乎可以肯定。神经网络的基本理念是用大量简单（本质上相同）的组件创建一个灵活的“计算结构”，并让这个“结构”能够被逐步修改，以便从实例中学习。

在目前的神经网络中，人们基本上是使用微积分的思想 —— 应用于实数 —— 来做这种增量修改。但越来越清楚的是，拥有高精度的数字并不重要；即使用目前的方法，8 位或更少的数字可能也足够了。

像蜂窝自动机这样的计算系统，基本上是在许多单独的比特上并行操作的，如何做这种增量修改从来都不清楚，但没有理由认为它不可能。事实上，就像“2012 年深度学习的突破”一样，这种增量修改在更复杂的情况下可能比简单的情况下更容易。

神经网络 —— 也许有点像大脑 —— 被设定为拥有一个基本固定的神经元网络，被修改的是它们之间连接的强度（“重量”）。（也许至少在年轻的大脑中，大量的完全新的连接也可以增长。） 但是，虽然这对生物学来[illegible]否是实现我们所需功能的最佳

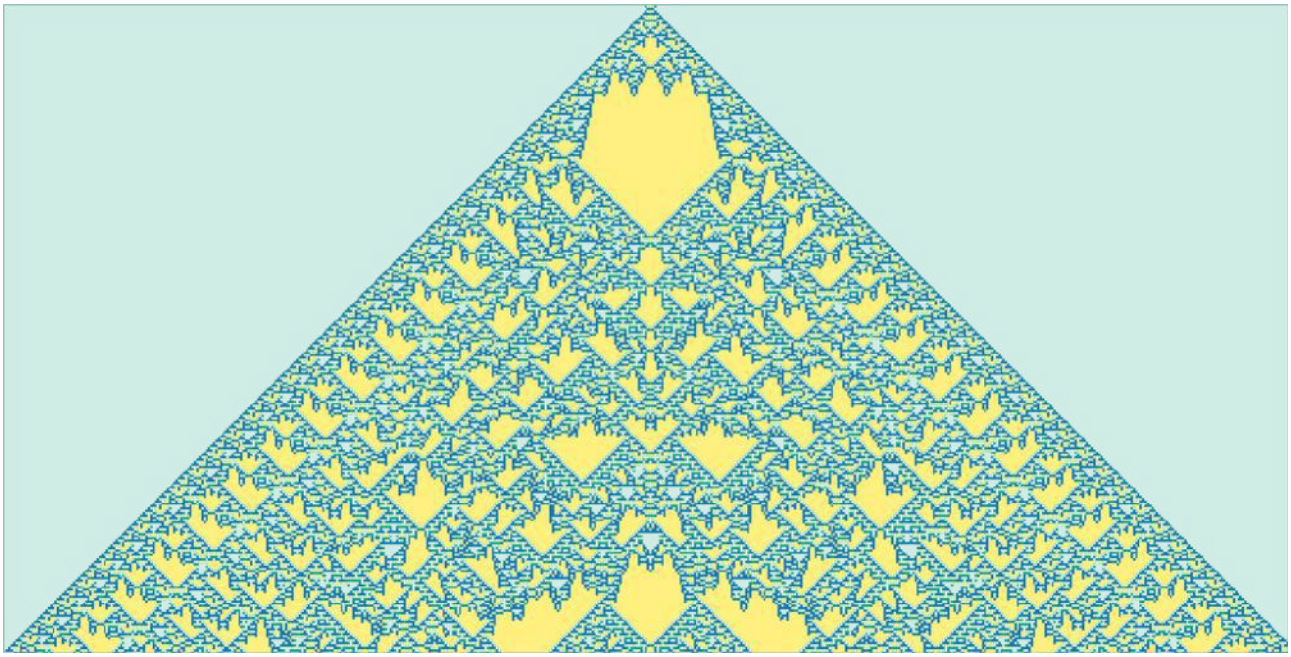
方式。而涉及渐进式网络重写的东西（也许让人想起我们的物理项目）最终可能会更好。

但即使在现有的神经网络框架内，目前也有一个关键的限制：现在的神经网络训练从根本上说是连续的，每一批例子的效果都被传播回来以更新权重。事实上，就目前的计算机硬件而言 —— 即使考虑到 GPU —— 在训练期间，神经网络的大部分时间都是“闲置”的，每次只有一个部分被更新。从某种意义上说，这是因为我们目前的计算机往往有独立于 CPU（或 GPU）的内存。但在大脑中，这大概是不同的 —— 每一个“记忆元素”（即神经元）也是一个潜在的活跃的计算元素。如果我们能够以这种方式设置我们未来的计算机硬件，就有可能更有效地进行训练。

“当然，一个足够大的网络可以做任何事情！”

像 ChatGPT 这样的能力似乎令人印象深刻，人们可能会想象，如果人们能够“继续下去”，训练越来越大的神经网络，那么它们最终将能够“做任何事情”。如果人们关注的是那些容易被人类直接思考的事物，那么很有可能是这样的。但是，过去几百年科学的教训是，有些东西可以通过形式化的过程来计算出来，但并不容易被人类的直接思维所获得。

非琐碎的数学就是一个大例子。但一般的情况其实是计算。而最终的问题是计算的不可还原性现象。有一些计算，人们可能认为需要很多步骤才能完成，但事实上可以“简化”为相当直接的东西。但计算的不可简化性的发现意味着这并不总是有效的。相反，有些过程 —— 可能就像下面这个过程 —— 要弄清楚其中发生了什么，必须要跟踪每个计算步骤进行追踪：



我们通常用大脑做的那些事情，大概是专门为避免计算的不可还原性而选择的。在一个人的大脑中做数学需要特别的努力。而且，在实践中，仅仅在一个人的大脑中“思考”任何非微观程序的操作步骤，在很大程度上是不可能的。

当然，为此我们有计算机。有了计算机，我们可以很容易地做很长的、计算上不可简化的事情。而关键的一点是，这些事情一般来说没有捷径。

是的，我们可以记住很多关于在某个特定计算系统中发生的具体例子。也许我们甚至可以看到一些（“计算上可还原的”）模式，使我们可以做一点概括。但问题是，计算上的不可还原性意味着我们永远无法保证意外不会发生 —— 只有通过明确地进行计算，你才能知道在任何特定情况下实际发生了什么。

最后，在可学习性和计算的不可重复性之间存在着一种基本的紧张关系。学习实际上是通过利用规则性来压缩数据。但计算上的不可复制性意味着最终对可能存在的规律性有一个限制。

作为一个实际问题，我们可以想象将一些小的计算设备 —— 如蜂窝自动机或图灵机 —— 构建成像神经网络这样的可训练系统。而且，这种设备确实可以作为神经网的好“工具”，就像 Wolfram|Alpha 可以作为 ChatGPT 的好工具。但计算的不可简化性意味着我们不能指望“进入”这些设备并让它们学习。

或者换句话说，在能力和可训练性之间有一个最终的权衡：你越想让一个系统“真正利用”它的计算能力，它就越会显示出计算的不可复制性，它的可训练性就越低。而它越是从根本上可训练，它就越不能做复杂的计算

（对于目前的 ChatGPT 来说，情况实际上要极端得多，因为用于生成每个输出符号的神经网络是一个纯粹的“前馈”网络，没有循环，因此没有能力做任何具有非复杂“控制流”的计算）。

当然，人们可能会问，能够做不可还原的计算是否真的很重要。事实上，在人类历史的大部分时间里，这并不特别重要。但我们的现代技术世界是建立在至少使用数学计算的工程之上的，而且越来越多地使用更普遍的计算。如果我们看一下自然界，它充满了不可简化的计算 —— 我们正在慢慢理解如何模仿并用于我们的技术目的。

是的，一个神经网络当然可以注意到自然世界中的各种规律性，而我们也可能很容易通过“无助的人类思维”注意到这些规律性。但是，如果我们想要解决属于数学或计算科学范畴的事情，神经网络是无法做到的——除非它有效地“作为工具”使用一个“普通”的计算系统。

但是，这一切都有一些潜在的混淆之处。在过去，有很多任务——包括写文章——我们认为对计算机来说“从根本上说太难了”。而现在我们看到这些任务是由 ChatGPT 等完成的，我们倾向于突然认为计算机一定是变得更加强大了，特别是超越了它们已经基本能够做到的事情（比如逐步计算蜂窝自动机等计算系统的行为）。

但这并不是正确的结论。计算上不可还原的过程仍然是计算上不可还原的，而且对计算机来说仍然是根本性的困难——即使计算机可以轻易地计算它们的单个步骤。相反，我们应该得出的结论是，我们人类可以做的，但我们不认为计算机可以做的任务，比如写文章，实际上在某种意义上比我们想象的更容易计算。

换句话说，神经网络之所以能够成功地写出一篇文章，是因为写一篇文章被证明是一个比我们想象的“计算上更浅”的问题。从某种意义上说，这使我们更接近于“拥有一种理论”，即我们人类是如何做到像写文章这样的事情的，或在一般情况下处理语言。

如果你有一个足够大的神经网络，那么，是的，你可能能够做任何人类能够轻易做到的事情。但是，你不会捕捉到自然界一般能做的事情——或者我们从自然界额外的工具能做的事情。而正是这些工具的使用——无

论是实用的还是概念性的 —— 使得我们在近几个世纪里能够超越“纯粹的无助的人类思维”所能达到的界限，并为人类的目的捕捉到物理和计算宇宙中的更多东西。

— 6 —

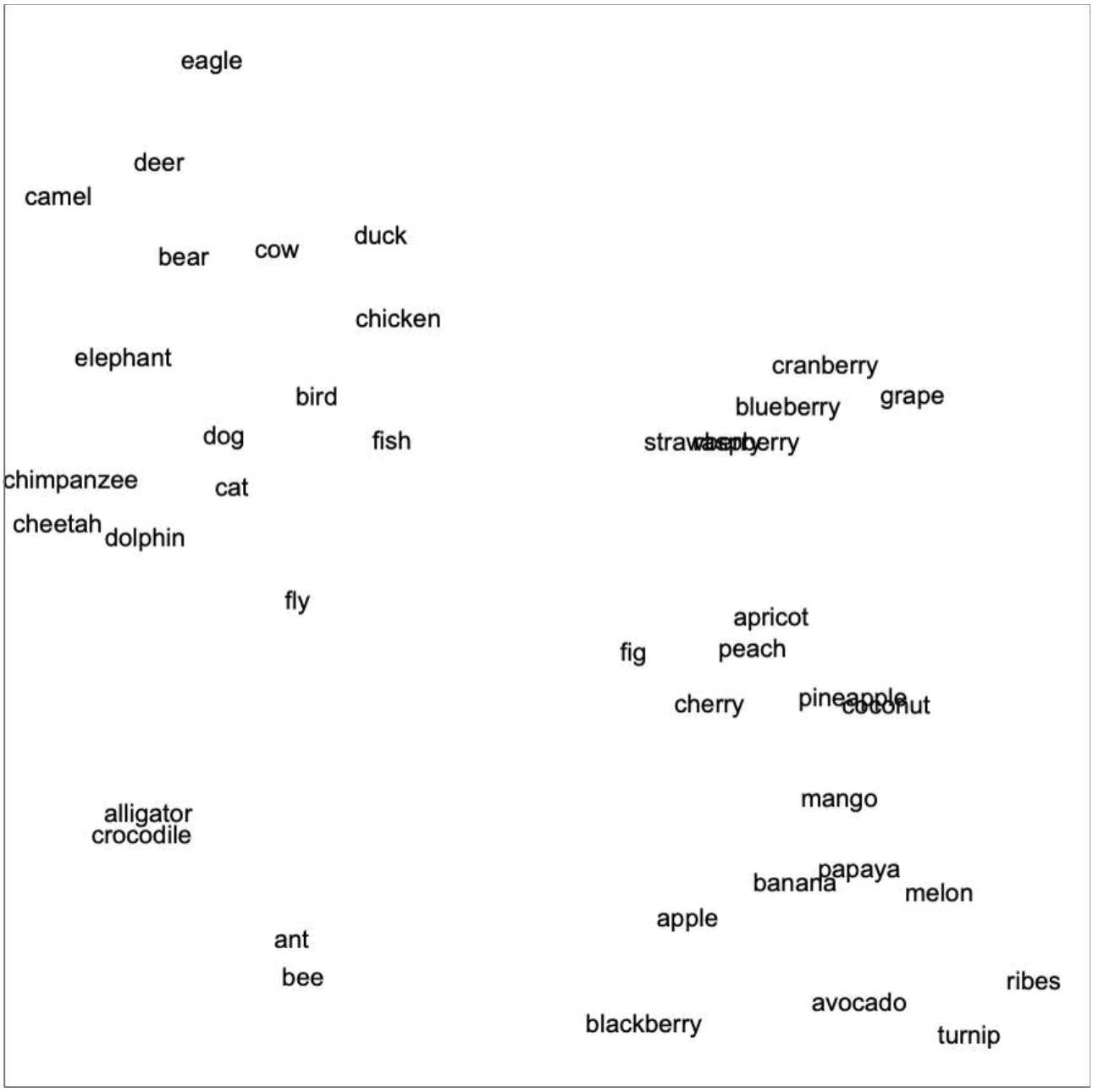
嵌入的概念

神经网络 —— 至少在它们目前的设置中 —— 从根本上说是基于数字的。因此，如果我们要用它们来处理像文本这样的东西，我们就需要一种方法来用数字表示我们的文本。

当然，我们可以开始（基本上就像 ChatGPT 那样）为字典中的每个词分配一个数字。但是，有一个重要的想法 —— 例如，它是 ChatGPT 的核心 —— 超出了这个范围。这就是“嵌入”的概念。我们可以把嵌入看作是一种尝试用数字阵列来表示事物“本质”的方式 —— 其特性是“附近的事物”由附近的数字来表示。

因此，举例来说，我们可以把一个词的嵌入看作是试图在一种“意义空间”中排列词语，在这个空间中，以某种方式“在意义上接近”的词语在嵌入中出现。实际使用的嵌入 —— 例如在 ChatGPT 中 —— 往往涉及大量

的数字列表。但是如果我们把它投射到二维空间，我们就可以显示出嵌入的单词是如何排列的例子：



而且，是的，我们看到的东西在捕捉典型的日常印象方面做得非常好。但是，我们怎样才能构建这样一个嵌入呢？大致的想法是查看大量的文本（这里是来自网络的 50 亿个词），然后看不同的词出现的“环境”有多相似。因此，例如，“alligator”和“crocodile”经常会在其他类似的句子中互换出

现，这意味着它们在嵌入中会被放在附近。但是 “萝卜” 和 “老鹰” 不会出现在其他类似的句子中，所以它们在嵌入中会被放在很远的地方。

但是，如何使用神经网络实际实现这样的东西呢？让我们先来讨论一下不是针对单词的嵌入，而是针对图像的嵌入。我们想找到某种方法，通过数字列表来描述图像，使 “我们认为相似的图像” 被分配到相似的数字列表中。

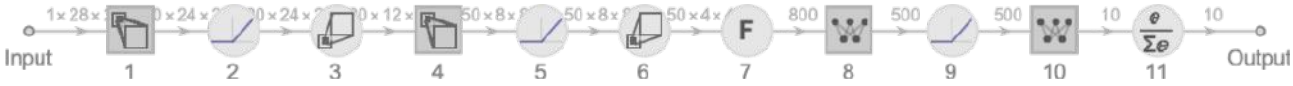
我们如何判断我们是否应该 “认为图像相似”？好吧，如果我们的图像是，例如，手写的数字，我们可能会 “认为两个图像是相似的”，如果它们是相同的数字。早些时候，我们讨论了一个被训练来识别手写数字的神经网络。我们可以认为这个神经网络被设置成在其最终输出中把图像放入 10 个不同的仓，每个数字一个仓。

但是，如果我们在做出 “这是一个 ‘4’” 的最终决定之前，“拦截” 神经网络内部发生的事情呢？我们可能会想到，在神经网络中，有一些数字将图像描述为 “大部分是 4，但有一点是 2” 或类似的情况。而我们的想法是挑选出这样的数字作为嵌入的元素。

所以这里有一个概念。我们不是直接试图描述 “什么图像在什么其他图像附近”，而是考虑一个定义明确的任务（在这种情况下是数字识别），我们可以获得明确的训练数据 —— 然后利用这样一个事实，即在做这个任务时，神经网络隐含地要做出相当于 “接近度决定” 的决定。因此，我们不需要明确地谈论 “图像的接近性”，而只是谈论一个图像代表什么数字

的具体问题，然后我们“把它留给神经网络”来隐含地决定这意味着什么“图像的接近性”。

那么，这对数字识别网络来说是如何更详细地工作的呢？我们可以认为这个网络是由 11 个连续的层组成的，我们可以用图标来概括它（激活函数显示为独立的层）：



在开始时，我们向第一层输入实际的图像，用像素值的二维阵列表示。在最后一层，我们得到了一个由 10 个值组成的数组，我们可以认为这表示网络对图像对应于 0 到 9 的每个数字的“确定程度”。

输入图像（手写的 4），最后一层的神经元的值就是：

$$\{1.42071\times10^{-22}, 7.69857\times10^{-14}, 1.9653\times10^{-16}, 5.55229\times10^{-21}, 1.,$$
$$8.33841\times10^{-14}, 6.89742\times10^{-17}, 6.52282\times10^{-19}, 6.51465\times10^{-12}, 1.97509\times10^{-14}\}$$

换句话说，神经网络此时已经“非常确定”这个图像是 4，为了实际得到输出“4”，我们只需挑选出数值最大的神经元的位置。

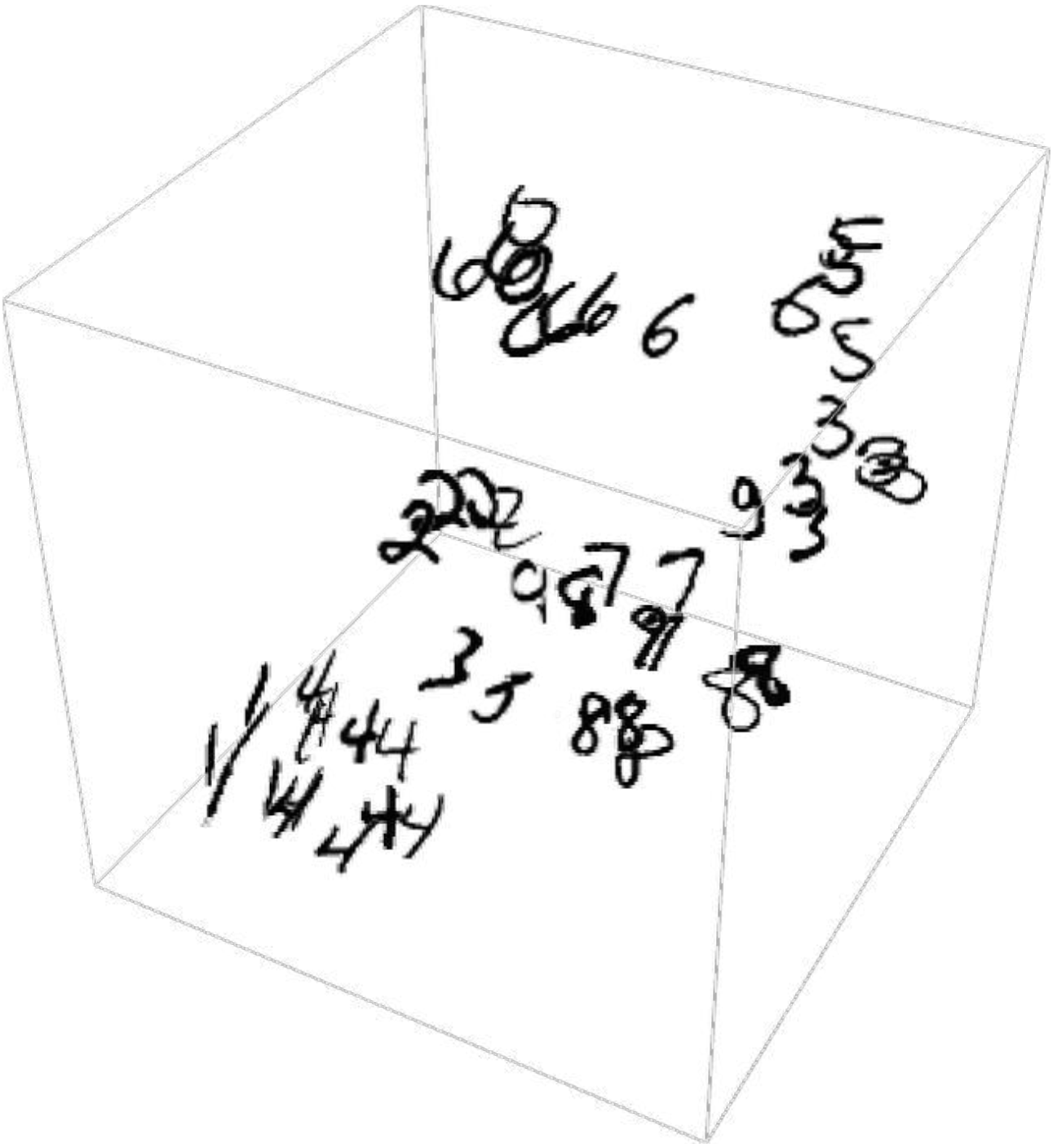
但是，如果我们再往前看一步呢？网络中的最后一个操作是一个所谓的 softmax，它试图“强制确定”。但在这之前，神经元的值是：

$$\{-26.134, -6.02347, -11.994, -22.4684,$$
$$24.1717, -5.94363, -13.0411, -17.7021, -1.58528, -7.38389\}$$

代表 “4” 的神经元仍然有最高的数值。但在其他神经元的数值中也有信息。我们可以期望这个数字列表在某种意义上可以用来描述图像的 “本质”，从而提供我们可以用作嵌入的东西。因此，例如，这里的每一个 4 都有一个稍微不同的 “签名”（或 “特征嵌入”） —— 都与 8 的非常不同：

在这里，我们基本上是用 10 个数字来描述我们的图像特征。但通常情况下，使用比这更多的数字会更好。例如，在我们的数字识别网络中，我们可以通过挖掘前一层得到一个 500 个数字的阵列。而这可能是一个合理的数组，作为 “图像嵌入” 使用。

如果我们想对手写数字的 “图像空间” 进行明确的可视化，我们需要 “降低维度”，有效地将我们得到的 500 维向量投射到，例如，三维空间：



我们刚刚谈到为图像创建一个特征（从而嵌入），有效地基于识别图像的相似性，确定（根据我们的训练集）它们是否对应于同一个手写数字。如果我们有一个训练集，比如说，确定每张图片属于 5000 种常见类型的物体（猫、狗、椅子……），我们就可以更普遍地对图片做同样的事情。

通过这种方式，我们可以制作一个图像嵌入，它被我们对常见物体的识别所“锚定”，但然后根据神经网络的行为“围绕它进行概括”。关键是，只要这种行为与我们人类感知和解释图像的方式相一致，这将最终成为一个“对我们来说是正确的”的嵌入，并在实践中做“类似人类判断”的任务时有用。

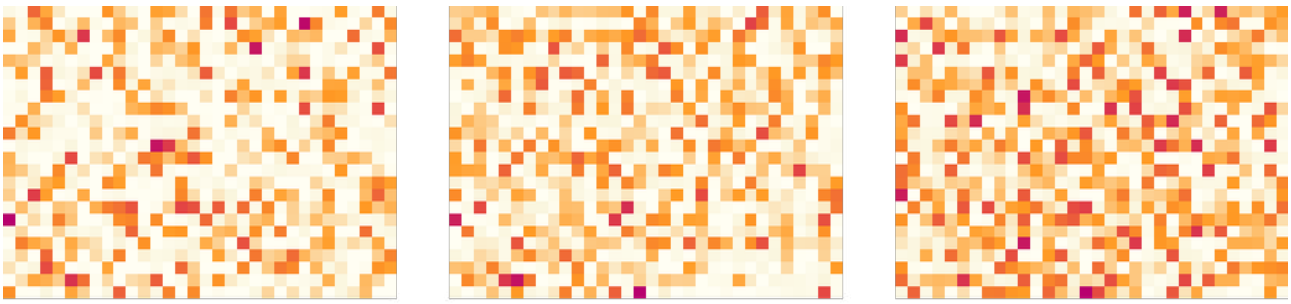
好吧，那么我们如何遵循同样的方法来寻找单词的嵌入呢？关键是要从一个我们可以随时进行训练的关于单词的任务开始。而标准的任务是“单词预测”。假设我们得到了“the cat”。基于一个大型的文本语料库（比如说，网络上的文本内容），可能“填空”的不同单词的概率是多少？或者说，给定“__ 黑 _”，不同的“侧翼词”的概率是多少？

我们如何为神经网络设置这个问题？归根结底，我们必须用数字来表述一切。做到这一点的一个方法就是为英语中 5 万个左右的常用词中的每一个分配一个独特的数字。因此，例如，“the”可能是 914，而“cat”（前面有一个空格）可能是 3542。（这些是 GPT-2 所使用的实际数字。）所以对于“the _ cat”问题，我们的输入可能是{914, 3542}。输出应该是什么样子的呢？好吧，它应该是一个由 50000 个左右的数字组成的列表，有效地给出了每个可能的“填充”单词的概率。

再一次，为了找到一个嵌入，我们要在神经网络“达到结论”之前“拦截”它的“内部”—— 然后捡起在那里出现的数字列表，我们可以把它看作是“每个词的特征”。

好吧，那么这些表征是什么样子的呢？在过去的 10 年里，已经有一系列不同的系统被开发出来（word2vec, GloVe, BERT, GPT, ……），每一个都是基于不同的神经网络方法。但最终，所有这些系统都是通过数百到数千个数字的列表来描述单词的特征。

在它们的原始形式中，这些“嵌入向量”是相当无信息的。例如，这里是 GPT-2 产生的三个特定词的原始嵌入向量：



如果我们做一些事情，比如测量这些向量之间的距离，那么我们就可以发现像单词的“接近性”这样的东西。稍后我们将更详细地讨论我们可能认为这种嵌入的“认知”意义。但现在主要的一点是，我们有一种方法可以有效地将单词变成“神经网络友好”的数字集合。

但实际上，我们可以更进一步，不仅仅是用数字的集合来描述单词；我们还可以对单词的序列，或者整个文本块进行描述。在 ChatGPT 中，它就是这样处理事情的。

它把目前得到的文本，生成一个嵌入矢量来表示它。然后，它的目标是找到接下来可能出现的不同词汇的概率。它将其答案表示为一个数字列表，

汇的概率。

（严格地说，ChatGPT 不处理单词，而是处理“符号”（token）—— 方便的语言单位，可能是整个单词，也可能只是“pre”或“ing”或“ized”这样的片段。使用符号使 ChatGPT 更容易处理罕见的、复合的和非英语的词汇，有时，无论好坏，都可以发明新的词汇。）

— 8 —

ChatGPT 内部

好了，我们终于准备好讨论 ChatGPT 内部的内容了。是的，最终，它是一个巨大的神经网络 —— 目前是所谓的 GPT-3 网络的一个版本，有 1750 亿个权重。在许多方面，这是一个非常像我们讨论过的其他神经网络。但它是一个特别为处理语言问题而设置的神经网络。它最显著的特征是一个叫做“转化器”的神经网络架构。

在我们上面讨论的第一个神经网络中，任何给定层的每个神经元基本上都与前一层的每个神经元相连（至少有一些权重）。但是，如果一个人在处理具有特殊的、已知的结构的数据时，这种全连接的网络（大概）是过剩的。因此，例如，在处理图像的早期阶段，典型的做法是使用所谓的卷积神经网络（“convnets”），其中的神经元被有效地布置在一个类似于图像中的像素的网格上 —— 并且只与网格上附近的神经元相连。

变换器的想法是为构成一段文本的标记序列做一些至少有点类似的事情。但是，转化器并不只是在序列中定义一个可以有连接的固定区域，而是引入了“注意”的概念 —— 以及对序列的某些部分比其他部分更“注意”的概念。也许有一天，仅仅启动一个通用的神经网络并通过训练进行所有的定制是有意义的。但至少到现在为止，将事情“模块化”在实践中似乎是至关重要的，就像变压器那样，可能也像我们的大脑那样。

好吧，那么 ChatGPT（或者说，它所基于的 GPT-3 网络）实际上是做什么的？回想一下，它的总体目标是以“合理”的方式延续文本，基于它所看到的训练（包括从网络上查看数十亿页的文本等），所以在任何时候，它都有一定数量的文本，它的目标是为下一个要添加的标记提出适当的选择。

它的操作分为三个基本阶段：

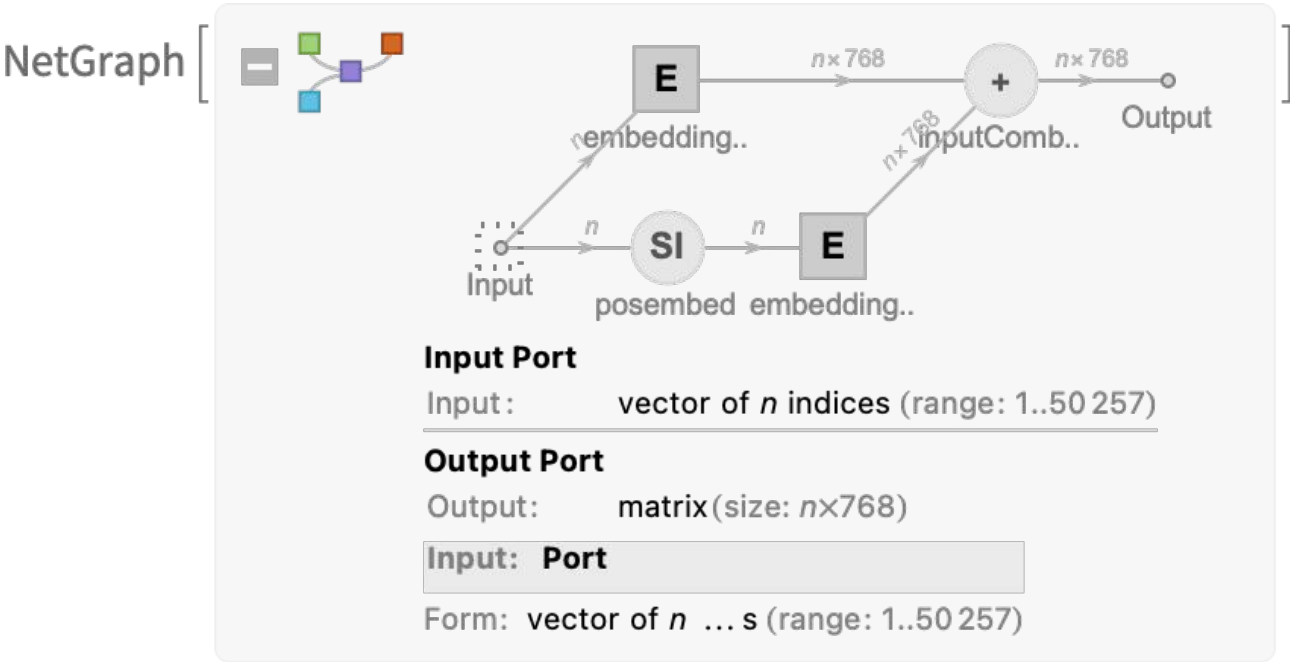
首先，它获取与迄今为止的文本相对应的标记序列，并找到代表这些标记的嵌入（即一个数字阵列）。其次，它以“标准的神经网络方式”对这一嵌入进行操作，数值“通过”网络中的连续层，产生一个新的嵌入（即一个新的数字阵列）。然后，它从这个数组的最后一部分，生成一个大约 50,000 个值的数组，这些值变成了不同的可能的下一个标记的概率。

（而且，是的，恰好使用的标记的数量与英语中的常用词的数量相同，尽管只有大约 3000 个标记是整个单词，其余的是片段。）关键的一点是，这个管道的每一部分都是由一个神经网络实现的，其权重是由网络的端到端训练决

定的。换句话说，实际上，除了整体架构之外，没有任何东西是“明确设计的”；所有东西都是从训练数据中“学习”的。

然而，在架构的设置方式上有很多细节，反映了各种经验和神经网络的传说。而且，尽管这肯定是进入了杂草丛中，但我认为谈论其中的一些细节是有用的，尤其是为了了解建立像 ChatGPT 这样的东西所需要的东西。

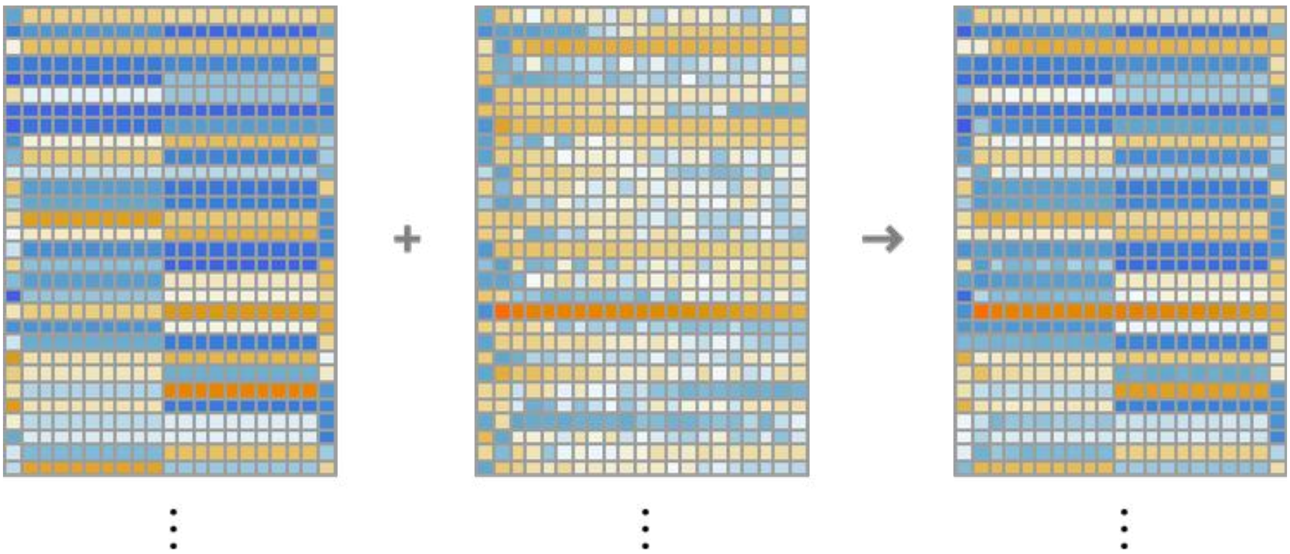
首先是嵌入模块。下面是 GPT-2 的 Wolfram 语言示意图：



输入是一个由 n 个标记组成的向量（如上一节所述，由 1 到 50,000 的整数表示）。这些标记中的每一个都被（通过单层神经网络）转换成一个嵌入向量（GPT-2 的长度为 768，ChatGPT 的 GPT-3 为 12,288）。同时，还有一个“二级路径”，它将标记的（整数）位置序列，并从这些整数中创建另一个嵌入向量。最后，来自令牌值和令牌位置的嵌入向量被加在一起——产生嵌入模块的最终嵌入向量序列。

为什么只是把令牌值和令牌位置的嵌入向量加在一起？我不认为这有什么特别的科学依据。只是各种不同的东西都被尝试过，而这是一个似乎有效的方法。这也是神经网络传说的一部分，从某种意义上说，只要你的设置是 “大致正确的”，通常就可以通过做充分的训练来确定细节，而不需要真正 “在工程层面上理解” 神经网络最终是如何配置它的。

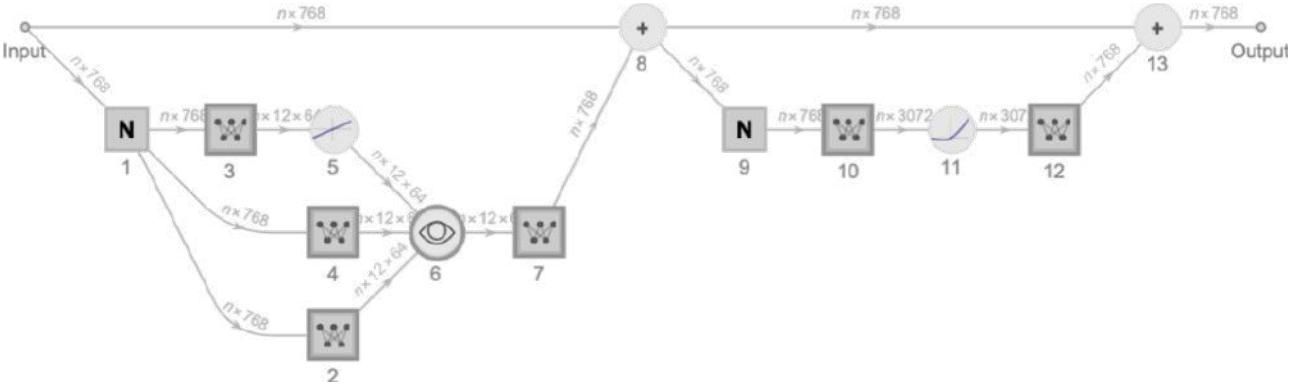
下面是嵌入模块所做的工作，对字符串 “hello hello hello hello hello bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye bye 2”：



每个标记的嵌入向量的元素都显示在页面下方，在整个页面上，我们首先看到的是 “hello” 的嵌入，然后是 “bye” 的嵌入。上面的第二个数组是位置嵌入 —— 其看起来有点随机的结构只是 “碰巧学到的”（在这种情况下是 GPT-2）。

好了，在嵌入模块之后，是转化器的 “主要事件”：一连串所谓的 “注意块”（GPT-2 为 12 个，ChatGPT 的 GPT-3 为 96 个）。这一切都很复杂

—— 让人想起典型的难以理解的大型工程系统，或者，生物系统。但无论如何，这里是一个单一的 “注意块” 的示意图（对于 GPT-2）：



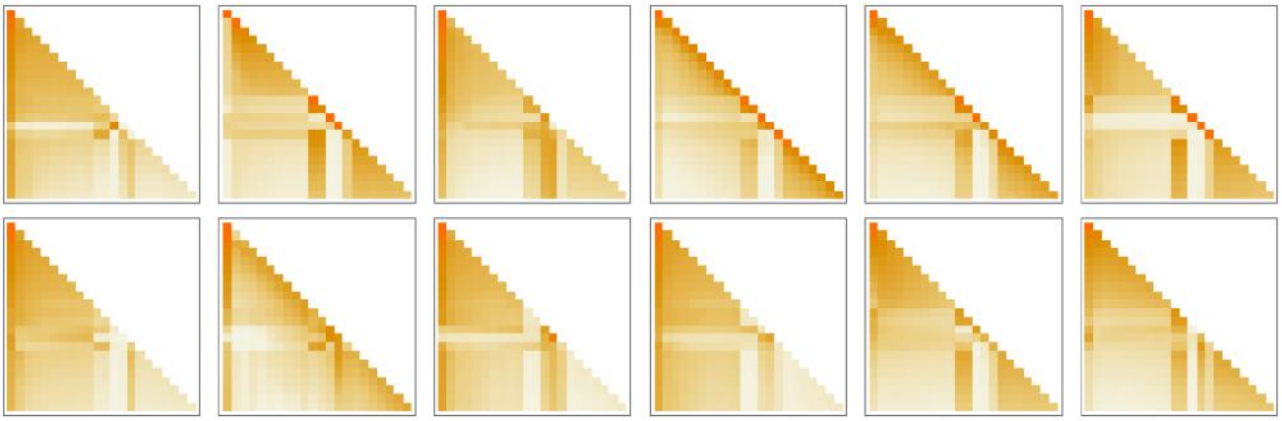
在每个这样的注意力块中，有一系列的 “注意力头”（GPT-2 有 12 个，ChatGPT 的 GPT-3 有 96 个） —— 每一个都是独立操作嵌入向量中的不同数值块的。（是的，我们不知道为什么分割嵌入向量是个好主意，或者它的不同部分有什么 “意义”；这只是 “被发现可行” 的事情之一）。

好吧，那么注意头是做什么的？基本上，它们是一种在标记序列中 “回顾” 的方式（即在迄今为止产生的文本中），并将过去的内容 “打包” 成有助于寻找下一个标记的形式。

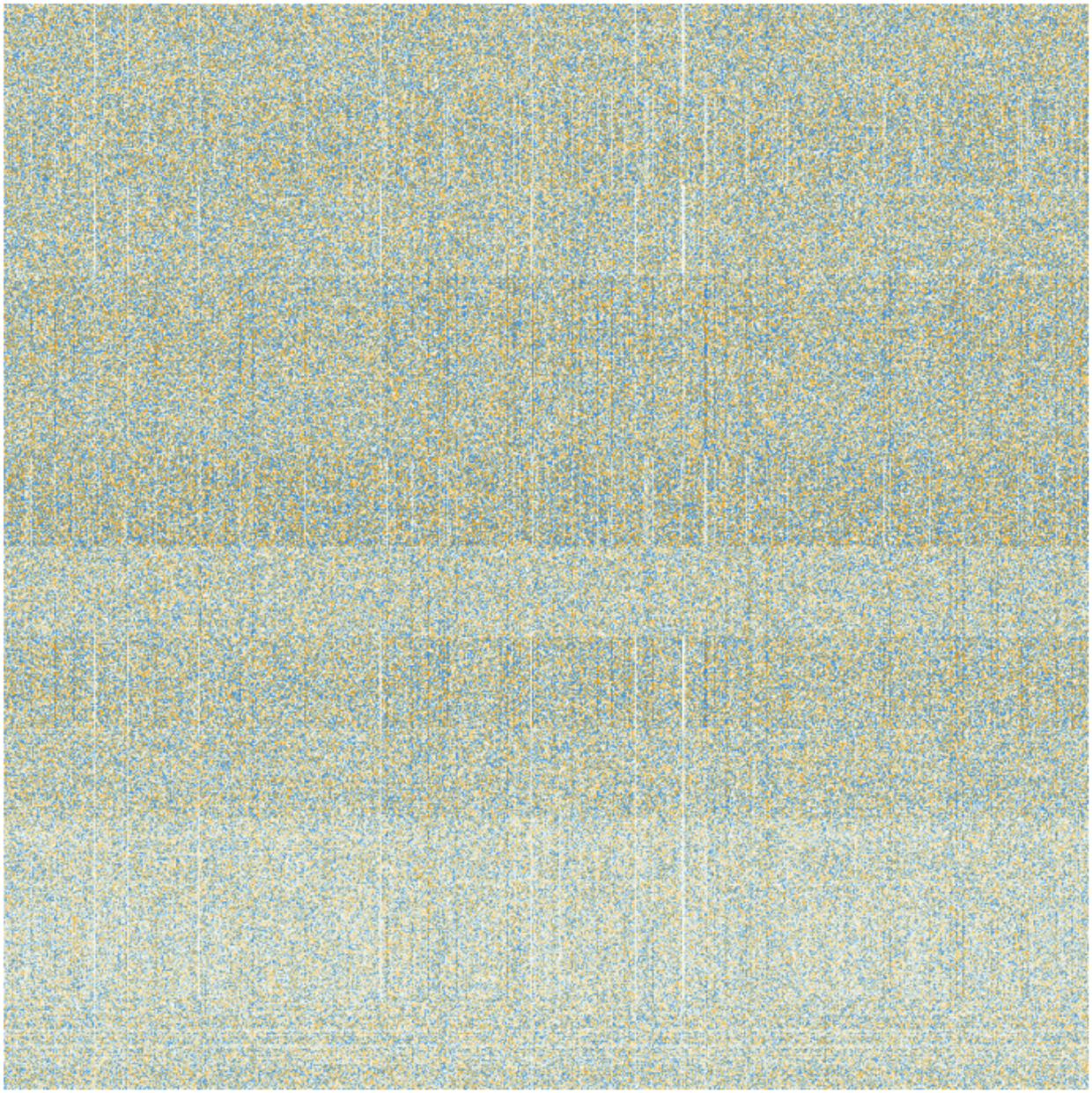
在上面的第一节中，我们谈到了使用 2-gram 概率来根据它们的直接前身来挑选单词。变换器中的 “注意” 机制所做的是允许 “注意” 甚至更早的词 —— 因此有可能捕捉到，比如说，动词可以指代在句子中出现在它们之前的许多词的名词的方式。

在更详细的层面上，注意力头所做的是以一定的权重重新组合与不同标记相关的嵌入向量中的大块。因此，例如，在第一个注意力区块中的 12 个

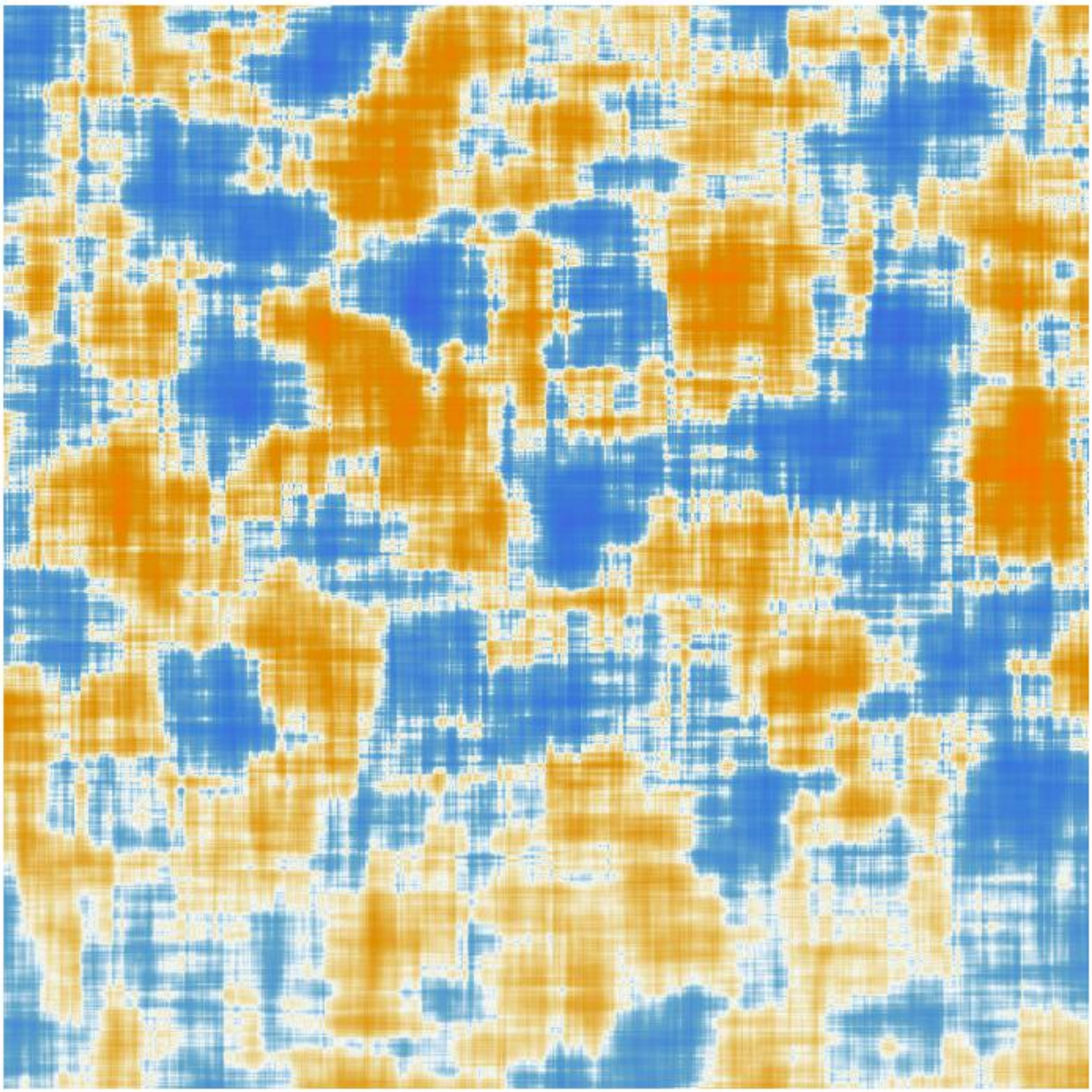
注意力头（在 GPT-2 中）对上面的 “hello, bye” 字符串有如下（“look-back-all-the-way-beginning-the-sequence-of-tokens”）模式的 “重组权值”：



在经过注意力头的处理后，产生的 “重新加权的嵌入向量”（GPT-2 的长度为 768，ChatGPT 的 GPT-3 的长度为 12288）被传递到一个标准的 “全连接” 神经网层。很难掌握这个层在做什么。但这里是它使用的 768×768 权重矩阵的图（这里是 GPT-2）：



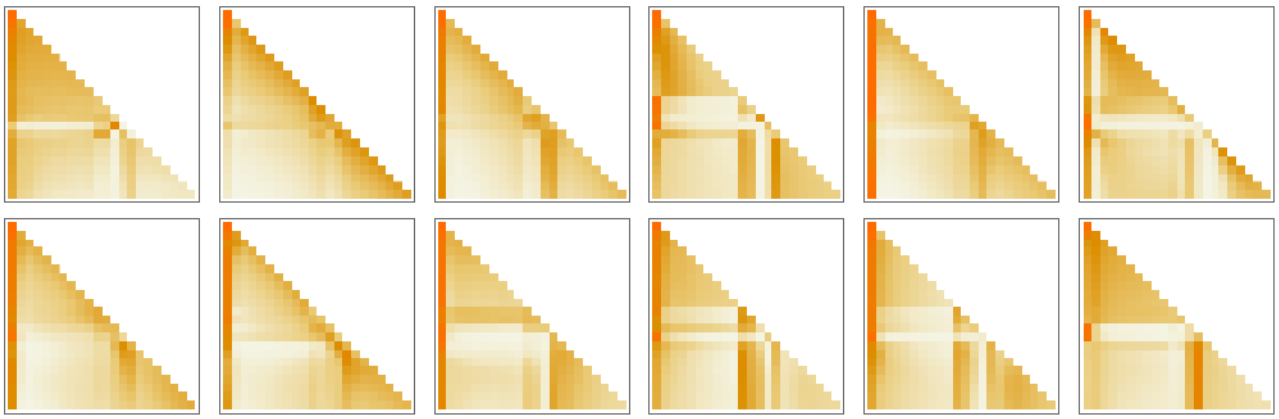
采用 64×64 的移动平均数，一些（随机漫步式的）结构开始出现：



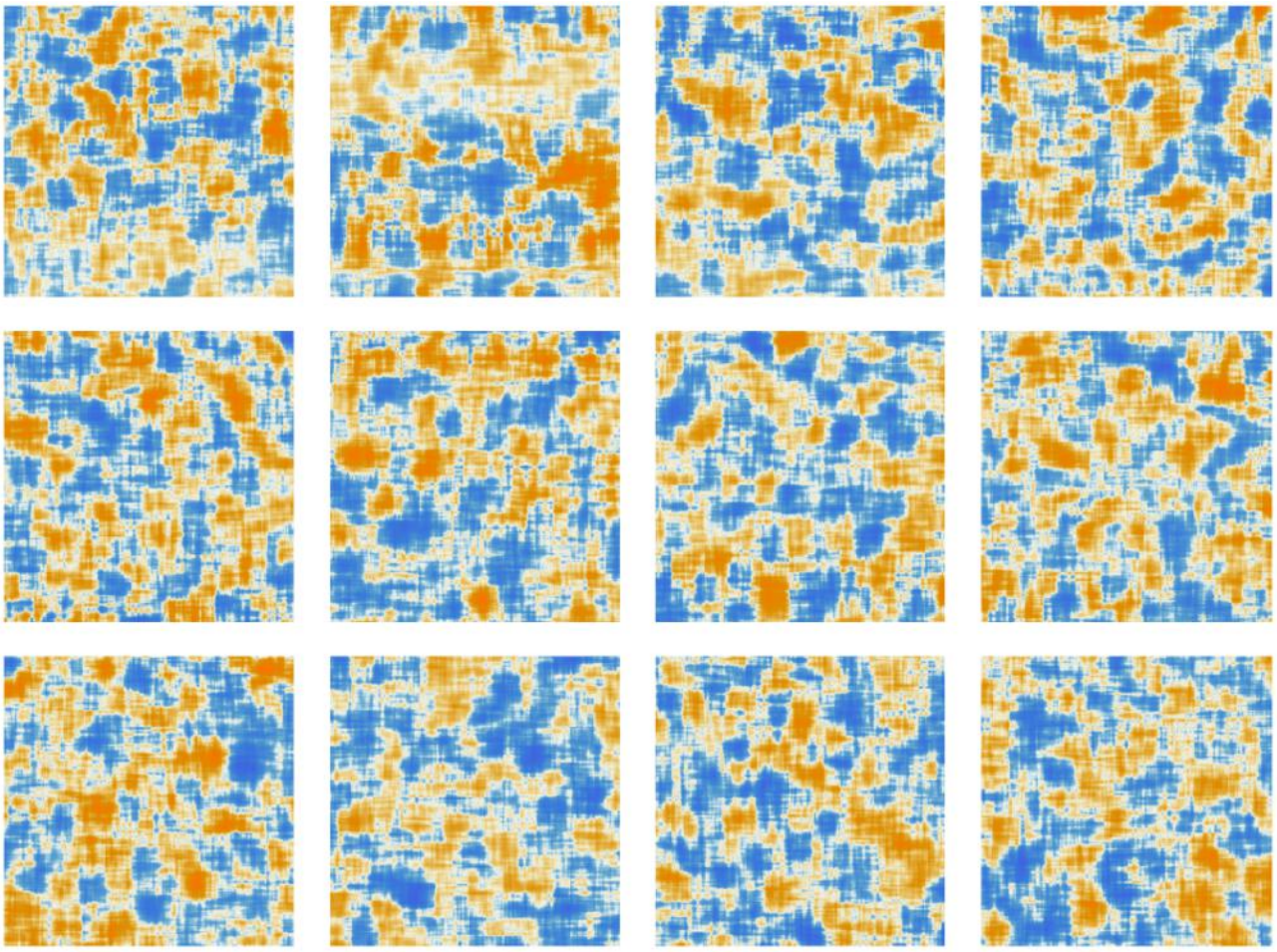
是什么决定了这种结构？最终，它可能是人类语言特征的一些“神经网络编码”。但到现在为止，这些特征可能是什么还很不清楚。实际上，我们正在“打开 ChatGPT 的大脑”（或至少是 GPT-2），并发现，是的，里面很复杂，而且我们不了解它 —— 尽管最终它产生了可识别的人类语言。

好吧，在经历了一个注意力区块之后，我们得到了一个新的嵌入向量 —— 然后它又被连续地传递到其他的注意力区块中（GPT-2 共有 12 个；

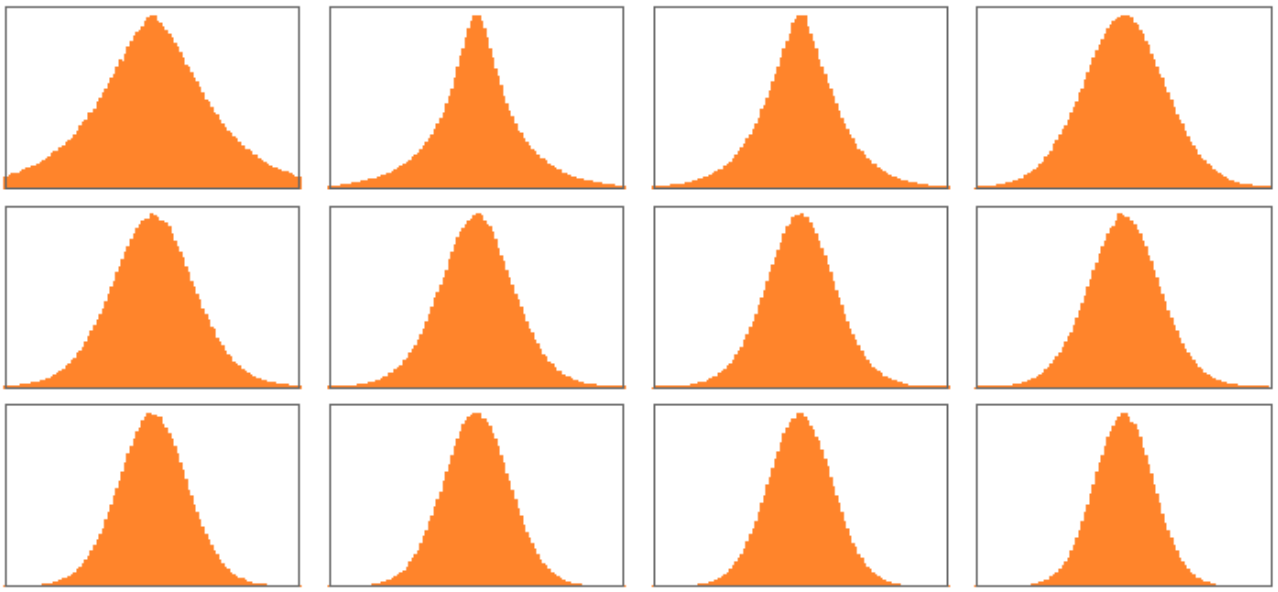
GPT-3 有 96 个）。每个注意力区块都有自己特定的“注意力”和“完全连接”权重模式。这里是 GPT-2 的“你好，再见”输入的注意权重序列，用于第一个注意头（attention head）：



这里是全连接层的（移动平均）“矩阵”：



奇怪的是，尽管这些“权重矩阵”在不同的注意力块中看起来很相似，但权重的大小分布可能有些不同（而且不总是高斯的）：



那么，在经历了所有这些注意力区块之后，转化器的净效果是什么？从本质上讲，它是将原始的符号序列的嵌入集合转化为最终的集合。而 ChatGPT 的具体工作方式是在这个集合中提取最后一个嵌入，并对其进行“解码”，以产生一个关于下一个标记应该是什么的概率列表。

这就是 ChatGPT 的概要内容。它可能看起来很复杂（尤其是因为它有许多不可避免的、有点武断的 “工程选择”），但实际上，所涉及的最终元素非常简单。因为最终我们要处理的只是一个由“人工神经元”组成的神经网络，每个神经元都在进行简单的操作，即接受一组数字输入，然后将它们与某些权重相结合。

ChatGPT 的原始输入是一个数字数组（到目前为止符号的嵌入向量），当 ChatGPT“运行” 以产生一个新的符号时，所发生的只是这些数字 “通过”

神经网的各层，每个神经元“做它的事”，并将结果传递给下一层的神经元。没有循环或“回头”。一切都只是通过网络“前馈”。

这是一个与典型的计算系统——如图灵机——非常不同的设置，在图灵机中，结果是由相同的计算元素反复“再处理”的。在这里，至少在生成一个特定的输出符号时，每个计算元素（即神经元）只被使用一次。

但在某种意义上，即使在 ChatGPT 中，仍然有一个重复使用计算元素的“外循环”。因为当 ChatGPT 要生成一个新的标记时，它总是“读取”（即作为输入）它之前的整个标记序列，包括 ChatGPT 自己之前“写”的标记。我们可以认为这种设置意味着 ChatGPT——至少在其最外层——涉及到一个“反馈循环”，尽管在这个循环中，每一次迭代都明确地显示为一个出现在其生成的文本中的标记。

但让我们回到 ChatGPT 的核心：反复用于生成每个标记的神经网络。在某种程度上，它非常简单：一整个相同的人工神经元的集合。网络的某些部分只是由（“完全连接”）的神经元层组成，其中某一层的每个神经元都与前一层的每个神经元相连（有一定的权重）。但是，特别是它的变压器结构，ChatGPT 有更多的结构部分，其中只有不同层的特定神经元被连接。（当然，人们仍然可以说，“所有的神经元都是连接的”——但有些神经元的权重为零）。

此外，ChatGPT 中的神经网的某些方面并不是最自然地被认为是由“同质”层组成的。例如，正如上面的图标摘要所示，在一个注意力区块中，[illegible]”，然后每个拷贝经过不同的

“处理路径”，可能涉及不同数量的层，然后才重新组合。但是，虽然这可能是对正在发生的事情的一种方便的表述，但至少在原则上总是可以考虑“密集地填入”层，但只是让一些权重为零。

如果我们看一下 ChatGPT 的最长路径，大约有 400 个（核心）层参与其中 —— 在某些方面不是一个巨大的数字。但是有数以百万计的神经元 —— 总共有 1750 亿个连接，因此有 1750 亿个权重。需要认识到的一点是，每当 ChatGPT 生成一个新的令牌时，它都要进行涉及这些权重中每一个的计算。

在实现上，这些计算可以“按层”组织成高度并行的阵列操作，可以方便地在 GPU 上完成。但是，对于产生的每一个标记，仍然要进行 1750 亿次计算（最后还要多一点） —— 因此，是的，用 ChatGPT 生成一个长的文本需要一段时间，这并不令人惊讶。

但最终，最了不起的是，所有这些操作 —— 它们各自都很简单 —— 能够以某种方式共同完成如此出色的“类似人类”的文本生成工作。必须再次强调的是，（至少到目前为止，我们知道）没有任何“最终的理论理由”来解释这样的工作。事实上，正如我们将要讨论的那样，我认为我们必须把这看作是一个潜在的令人惊讶的科学发现：在像 ChatGPT 这样的神经网络中，有可能捕捉到人类大脑在生成语言方面的本质。

— 9 —

ChatGPT 的训练

好了，现在我们已经给出了 ChatGPT 建立后的工作概要。但它是如何建立的呢？其神经网络中的 1750 亿个权重是如何确定的？基本上，它们是非常大规模的训练的结果，基于一个巨大的文本语料库 —— 网络上的、书中的等等 —— 由人类写的。

正如我们所说的，即使考虑到所有的训练数据，神经网络是否能够成功地产生“类似人类”的文本，这一点也不明显。而且，再一次，似乎需要详细的工程来实现这一目标。但 ChatGPT 的最大惊喜和发现是，它是可能的。实际上，一个“只有”1750 亿个权重的神经网络可以对人类所写的文本做出一个“合理的模型”。

在现代，有很多人类写的文本是以数字形式存在的。公共网络至少有几十亿人写的网页，总共可能有一万亿字的文本。如果包括非公开网页，这些数字可能至少要大 100 倍。到目前为止，已经有超过 500 万本数字化书籍可供使用（在曾经出版过的 1 亿本左右的书籍中），又有 1000 亿左右的文字。

作为个人比较，我一生中发表的材料总字数不到 300 万字，在过去 30 年[illegible]打了大约 5000 万字，在过去

几年中，我在直播中说了 1000 多万字。而且，是的，我将从所有这些中训练一个机器人）。

但是，好吧，鉴于所有这些数据，我们如何从中训练出一个神经网络呢？基本过程与我们在上面的简单例子中讨论的非常相似。你提出一批例子，然后你调整网络中的权重，使网络在这些例子上的误差（“损失”）最小。从错误中“反向传播”的主要问题是，每次你这样做，网络中的每个权重通常至少会有微小的变化，而且有大量的权重需要处理。（实际的“反向计算”通常只比正向计算难一个小常数）。

有了现代的 GPU 硬件，从成千上万的例子中并行计算出结果是很简单的。但是，当涉及到实际更新神经网络中的权重时，目前的方法要求我们基本上是一批一批地做。（是的，这可能是实际的大脑 —— 其计算和记忆元素的结合 —— 目前至少有一个架构上的优势）。

即使在我们之前讨论的看似简单的学习数字函数的案例中，我们发现我们经常不得不使用数百万个例子来成功训练一个网络，至少从头开始。那么，这意味着我们需要多少个例子来训练一个“类人语言”模型呢？似乎没有任何基本的“理论”方法可以知道。但是在实践中，ChatGPT 已经成功地在几千亿字的文本上进行了训练。

有些文本被多次输入，有些只有一次。但不知何故，它从它看到的文本中“得到了它需要的东西”。但是，考虑到需要学习的文本量，它应该需要多大的网络才能“学好”？同样，我们还没有一个基本的理论方法来说明。

最终 —— 我们将在下面进一步讨论 —— 人类语言大概有某种“总的算法内容”，以及人类通常用它说什么。但接下来的问题是，神经网络在实现基于该算法内容的模型时将会有多大的效率。我们也不知道 —— 尽管 ChatGPT 的成功表明它的效率还算不错。

最后我们可以注意到，ChatGPT 使用了几千亿个权重 —— 与它所获得的训练数据的总字数（或令牌）相比，它所做的事情是相当的。在某些方面，也许令人惊讶的是（尽管在 ChatGPT 的小型类似物中也有经验观察），似乎工作良好的“网络规模”与“训练数据的规模”如此相似。毕竟，这肯定不是说“在 ChatGPT 内”所有来自网络和书籍等的文本都被“直接存储”了。因为在 ChatGPT 里面的实际上是一堆数字 —— 精度略低于 10 位 —— 是对所有这些文本的总体结构的某种分布式编码。

换句话说，我们可以问人类语言的“有效信息含量”是什么，以及通常用它说什么。这里有语言实例的原始语料库。然后是 ChatGPT 的神经网络中的表述。这个表征很可能与“算法上最小”的表征相去甚远（我们将在下面讨论）。但它是一个很容易被神经网络使用的表征。在这种表示法中，训练数据的“压缩”程度似乎很低；平均而言，似乎只需要不到一个神经网络的权重就可以承载一个词的训练数据的“信息内容”。

当我们运行 ChatGPT 来生成文本时，我们基本上不得不使用每个权重一次。因此，如果有 n 个权重，我们有 n 个计算步骤要做 —— 尽管在实践中，许多步骤通常可以在 GPU 中并行完成。但是，如果我们需要大约 n 个字的训练数据来设置这些权重，那么从我们上面所说的，我们可以得出

结论，我们需要大约 n2 个计算步骤来进行网络训练 —— 这就是为什么，用目前的方法，人们最终需要谈论数十亿美元的训练工作。

— 10 —

基本训练之上

训练 ChatGPT 的大部分工作是向它“展示”大量来自网络、书籍等的现有文本。但事实证明，还有一个明显相当重要的部分。

一旦它完成了对所展示的原始语料库的“原始训练”，ChatGPT 内的神经网络就可以开始生成自己的文本，继续提示等。但是，虽然这样做的结果往往看起来很合理，但它们往往 —— 特别是对于较长的文本 —— 以往往相当非人类的方式“游离”。这不是人们可以轻易发现的，比如说，通过对文本做传统的统计。但这是实际阅读文本的人很容易注意到的东西。

构建 ChatGPT 的一个关键想法是，在“被动地阅读”网络等事物之后，还有一个步骤：让实际的人类主动与 ChatGPT 互动，看看它产生了什么，并在实际上给它反馈“如何成为一个好的聊天机器人”。

但神经网络如何使用这种反馈呢？第一步只是让人类对神经网络的结果进行评价。但随后又建立了另一个神经网络模型，试图预测这些评分。但现

在这个预测模型可以在原始网络上运行 —— 基本上就像一个损失函数，实际上是让该网络通过人类的反馈来 “调高”。而实践中的结果似乎对系统成功产生 “类似人类” 的输出有很大影响。

总的来说，有趣的是，“最初训练的” 网络似乎只需要很少的 “戳” 就能让它向特定的方向有用地发展。人们可能会认为，要让网络表现得像 “学到了新东西”，就必须运行训练算法，调整权重，等等。

但事实并非如此。相反，基本上只需要告诉 ChatGPT 一些东西，作为你所给的提示的一部分，然后它就可以在生成文本时成功地利用你告诉它的东西。我认为，这一点再次成为理解 ChatGPT “真正在做什么” 以及它与人类语言和思维结构的关系的一个重要线索。

这当然有一些类似于人类的东西：至少在它接受了所有的预训练之后，你可以告诉它一些东西，而它可以 “记住它” —— 至少 “足够长的时间” 来使用它生成一段文本。那么，在这样的情况下发生了什么？

可能是 “你可能告诉它的一切都已经在那里了” —— 你只是把它引向正确的地方。但这似乎并不靠谱。相反，似乎更有可能的是，是的，这些元素已经在那里了，但具体细节是由 “这些元素之间的轨迹” 这样的东西来定义的，这就是你告诉它的东西。

事实上，就像人类一样，如果你告诉它一些奇怪的、出乎意料的、完全不适合它所知道的框架的东西，它似乎并不能成功地 “整合” 这个。只有当

它基本上以一种相当简单的方式骑在它已经拥有的框架之上时，它才能“整合”它。

还值得再次指出的是，对于神经网络能够“接收”的东西，不可避免地存在“算法限制”。告诉它“浅层”的规则，如“这个到那个”，神经网络很可能能够很好地表示和再现这些规则 —— 事实上，它从语言中“已经知道”的东西会给它一个直接的模式来遵循。

但是，如果试图给它制定一个实际的“深度”计算规则，涉及许多潜在的不可简化的计算步骤，它就无法工作了。（记住，在每一步，它总是在其网络中“向前输送数据”；除了生成新的标记外，从不循环。）

当然，网络可以学习特定的“不可简化的”计算的答案。但只要有组合数的可能性，这种“查表式”的方法就不会奏效。因此，是的，就像人类一样，现在是时候让神经网络“伸出手来”，使用实际的计算工具了。（是的，Wolfram|Alpha 和 Wolfram 语言是唯一合适的，因为它们是为了“谈论世界上的事物”而建立的，就像语言模型的神经网络一样）。

— 11 —

是什么真正让 ChatGPT 工作?

人类的语言 —— 以及产生语言的思维过程 —— 似乎一直代表着一种复杂性的顶峰。事实上，人类的大脑 ——“仅”有 1000 亿个左右的神经元网络（也许还有 100 万亿个连接） —— 能够负责这项工作，似乎有些了不起。也许，人们可能会想象，大脑除了神经元网络之外还有其他东西，就像一些未被发现的物理学新层。

但现在通过 ChatGPT，我们得到了一个重要的新信息：我们知道，一个纯粹的人工神经网络，其连接数与大脑的神经元一样多，能够很好地生成人类语言，令人惊讶。

而且，是的，这仍然是一个庞大而复杂的系统 —— 其神经网络的权重与目前世界上的文字一样多。但在某种程度上，似乎仍然很难相信，语言的所有丰富性和它可以谈论的东西可以被封装在这样一个有限的系统中。

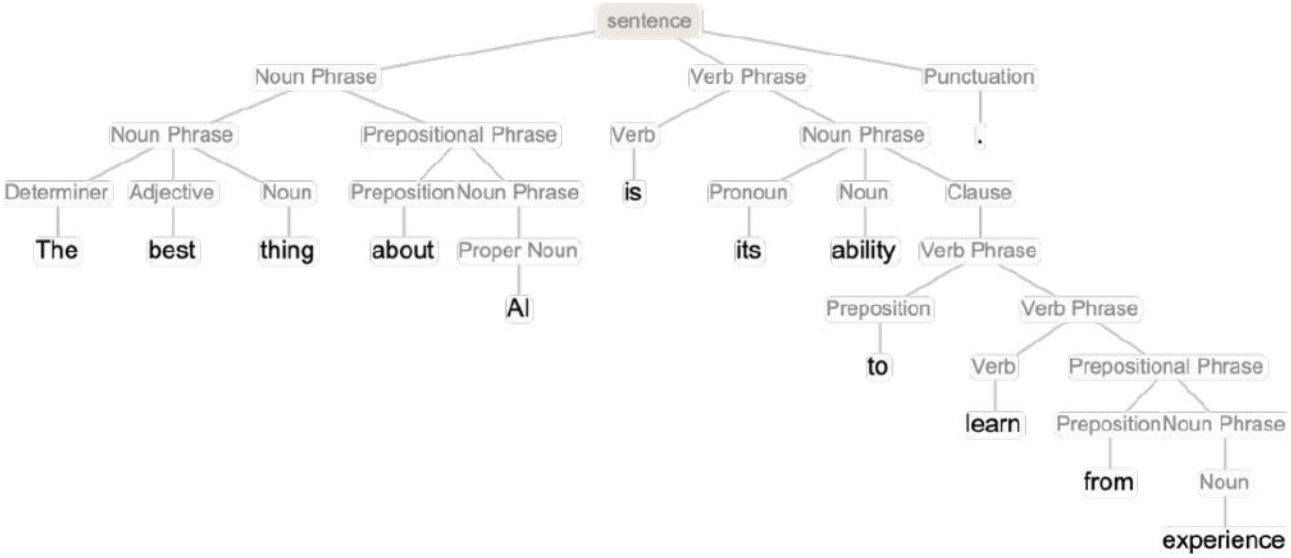
这其中的部分原因无疑是反映了一个无处不在的现象（这在第 30 条规则的例子中首次变得很明显），即计算过程实际上可以大大放大系统的表面复杂性，即使其基本规则很简单。但是，实际上，正如我们上面所讨论的，ChatGPT 中所使用的那种神经网络往往是专门用来限制这种现象的影响以及与之相关的计算的不可重复性的，以便使其训练更容易进行。

那么，像 ChatGPT 这样的东西是如何在语言方面走得如此之远的呢？我想，基本的答案是，语言在根本层面上比它看起来要简单得多。这意味着 ChatGPT —— 即使它的神经网络结构最终是简单的 —— 能够成功地“捕捉”人类语言的本质和背后的思维。此外，在其训练中，ChatGPT 以某种方式“隐含地发现”了语言（和思维）中的任何规律性，使其成为可能。

我认为，ChatGPT 的成功为我们提供了一个基本的和重要的科学证据：它表明我们可以期待有重大的新“语言法则”—— 以及有效的“思维法则”—— 在那里被发现。在 ChatGPT 中，作为一个神经网络，这些规律充其量是隐含的。但是，如果我们能以某种方式使这些定律明确化，就有可能以更直接、更有效和更透明的方式完成 ChatGPT 所做的各种事情。

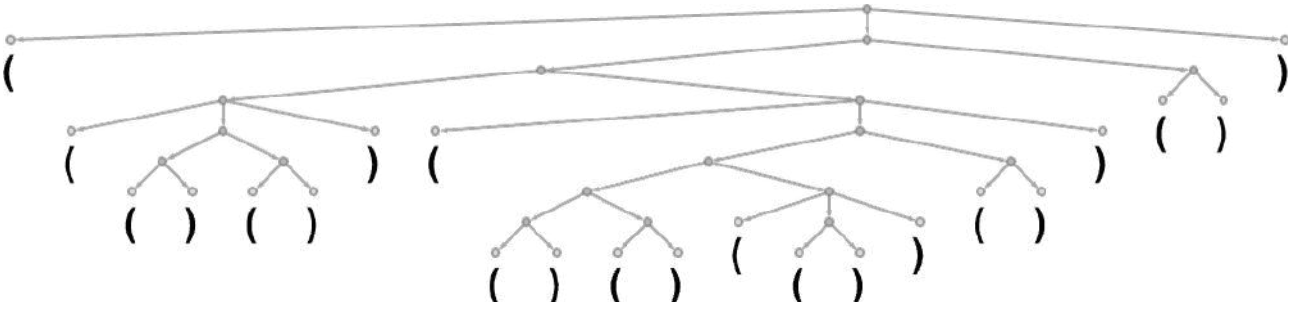
但是，好吧，那么这些法律可能是什么样的？最终，它们必须给我们提供某种语言 —— 以及我们用它说的东西 —— 如何组合的处方。稍后我们将讨论“观察 ChatGPT”如何能够给我们一些这方面的提示，以及我们从构建计算语言中了解到的情况如何提示我们前进的道路。但首先让我们来讨论两个长期以来为人所知的相当于“语言法则”的例子 —— 以及它们与 ChatGPT 的运作有何关系。

第一个是语言的语法。语言并不只是一个随机的词语组合。相反，对于不同种类的单词如何放在一起，有（相当）明确的语法规则：例如，在英语中，名词前面可以有形容词，后面可以有动词，但通常两个名词不能紧挨着。这样的语法结构可以（至少是近似地）被一套规则所捕获，这些规则定义了如何将相当于“解析树”的东西放在一起：



ChatGPT 对这种规则没有任何明确的 “知识”。但在训练中，它隐含地 “发现” 了这些规则，然后似乎很擅长遵循这些规则。那么，它是如何工作的呢？在一个 “大画面” 的层面上，这并不清楚。但是为了得到一些启示，看看一个更简单的例子也许会有启发。

考虑一种由（）和（）序列组成的 “语言”，其语法规定括号应该总是平衡的，如解析树所表示的那样：



我们能否训练一个神经网络来产生 “语法上正确的” 小括号序列？在神经网络中处理序列有多种方法，但让我们使用变换器网络，就像 ChatGPT 那样。给定一个简单的变换器网络，我们可以开始给它提供语法正确的小括号序列作为训练实例

一个微妙之处（实际上也出现在 ChatGPT 的人类语言生成中）是，除了我们的 “内容标记”（这里是 “（” 和 “）”），我们还必须包括一个 “结束” 标记，它的生成表明输出不应该再继续下去（即对于 ChatGPT 来说，我们已经到达了 “故事的终点”）。

如果我们只用一个有 8 个头的注意块和长度为 128 的特征向量来设置一个转换网（ChatGPT 也使用长度为 128 的特征向量，但有 96 个注意块，每个注意块有 96 个头），那么似乎不可能让它学会很多小括号语言。但是，如果有 2 个注意力头，学习过程似乎会收敛 —— 至少在给出 1000 万个左右的例子之后（而且，正如转化器网络所常见的那样，显示更多的例子似乎会降低其性能）。

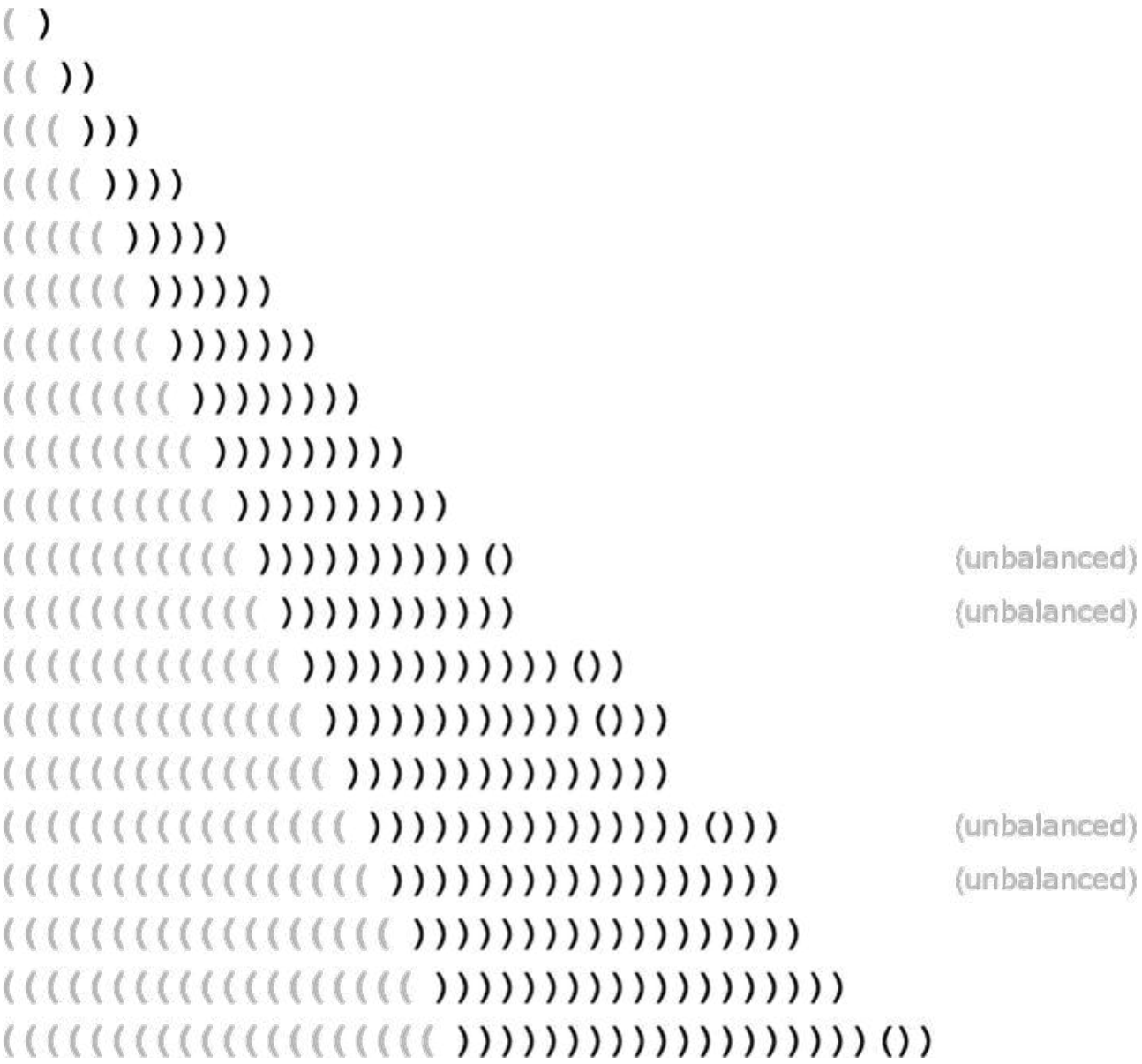
因此，对于这个网络，我们可以做 ChatGPT 的类似工作，并询问下一个标记应该是什么的概率 —— 在一个括号序列中：

((( ) ( ) (		
	(	46%
	)	54%
	End	0.038%

((( ) ( ))		
	(	51%
	)	15%
	End	34%

在第一种情况下，网络 “非常确定” 序列不能在这里结束 —— 这很好，因为如果它结束了，小括号就会留下不平衡。然而，在第二种情况下，它 “正确地认识到” 序列可以在这里结束，尽管它也 “指出” 有可能 “重新开始”，放下一个 “（”，估计后面还有一个 “）”。但是，哎呀，即使它有 40 万个左右经过艰苦训练的权重，它也说有 15% 的概率将 “）” 作为下一个[illegible]个不平衡的括号。

如果我们要求网络为逐渐变长的（）序列提供最高概率的完成度，我们会得到以下结果：



是的，在一定长度内，网络做得很好。但随后它就开始失败了。这是在神经网络（或一般的机器学习）的这种“精确”情况下看到的非常典型的事情。人类“一眼就能解决”的情况，神经网络也能解决。但是需要做一些“更多的算法”（例如明确地计算括号是否封闭）的情况，神经网络往往在

某种程度上是“计算上太浅”，无法可靠地做到。（顺便说一句，即使是目前完整的 ChatGPT 也很难正确匹配长序列中的括号）。

那么，这对像 ChatGPT 和像英语这样的语言的语法意味着什么呢？小括号语言是“朴素的”——而且更像是一个“算法的故事”。但在英语中，能够在局部选词和其他提示的基础上“猜测”什么是符合语法的，则要现实得多。

而且，是的，神经网络在这方面要好得多——尽管它可能会错过一些“形式上正确”的情况，而人类也可能错过。但主要的一点是，语言有一个整体的句法结构这一事实——以及它所暗示的所有规律性——在某种意义上限制了神经网络要学习的“程度”。一个关键的“类似自然科学”的观察是，像 ChatGPT 中的神经网络的转化器架构似乎能够成功地学习所有人类语言中似乎都存在（至少在某种程度上是近似的）的那种嵌套树状的句法结构。

句法提供了对语言的一种约束。但显然还有更多。像“好奇的电子吃鱼的蓝色理论”这样的句子在语法上是正确的，但并不是人们通常期望说的东西，而且如果 ChatGPT 生成它，也不会被认为是成功的——因为，嗯，以其中单词的正常含义，它基本上没有意义。

但是，是否有一个一般的方法来判断一个句子是否有意义？这方面没有传统的整体理论。但是，我们可以认为 ChatGPT 在接受了来自网络的数十亿（可能是有意义的）句子的训练之后，已经隐含地“发展了一套理论”。

这个理论可能是什么样的呢？好吧，有一个小小的角落，基本上两千年来一直为人所知，那就是逻辑。当然，在亚里士多德发现的 Syllogistic 形式中，逻辑基本上是一种说法，即遵循某些模式的句子是合理的，而其他的则不是。

因此，例如，说 “所有的 X 都是 Y，这不是 Y，所以它不是 X” 是合理的（正如 “所有的鱼都是蓝色的，这不是蓝色，所以它不是鱼”）。就像人们可以有点异想天开地想象亚里士多德通过（“机器学习式”）大量的修辞学例子来发现对偶逻辑一样，人们也可以想象在 ChatGPT 的训练中，它将能够通过查看网络上的大量文本等来 “发现对偶逻辑”。

（是的，虽然我们可以期待 ChatGPT 产生包含 “正确推论” 的文本，比如基于对偶逻辑，但当它涉及到更复杂的形式逻辑时，情况就完全不同了 —— 我认为我们可以期待它在这里失败，原因与它在小括号匹配中失败的原因相同）。

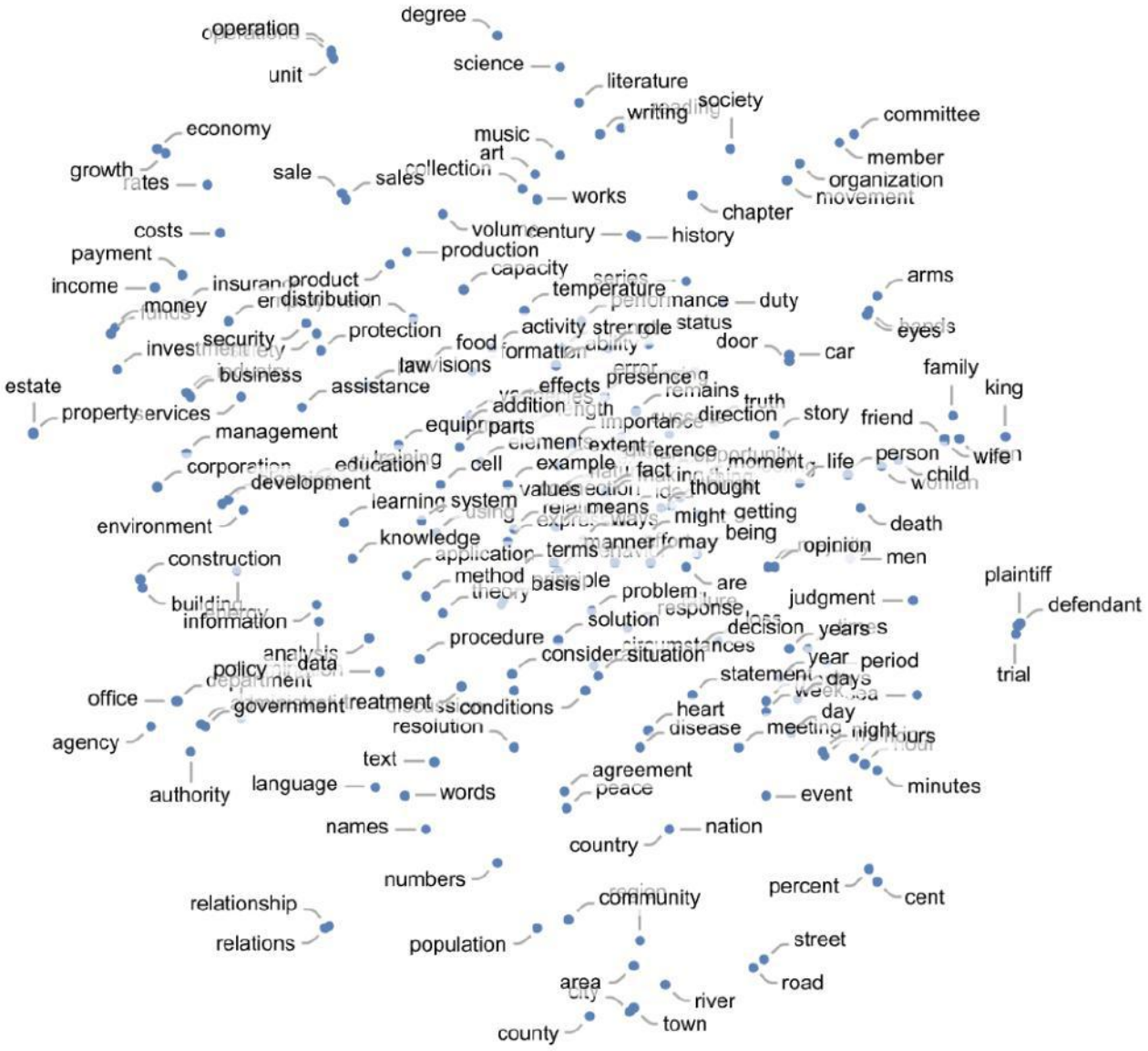
但除了逻辑这个狭隘的例子之外，对于如何系统地构建（或识别）甚至是合理的有意义的文本，又能说些什么呢？是的，有一些东西，如《疯狂的自由》，使用非常具体的 “短语模板”。但不知何故，ChatGPT 隐含着一种更普遍的方法。也许除了 “当你有 1750 亿个神经网络权重时，它就会以某种方式发生” 之外，对如何做到这一点没有什么可说的。但我强烈怀疑有一个更简单、更有力的故事。

— 12 —

意义空间和语义运动法则

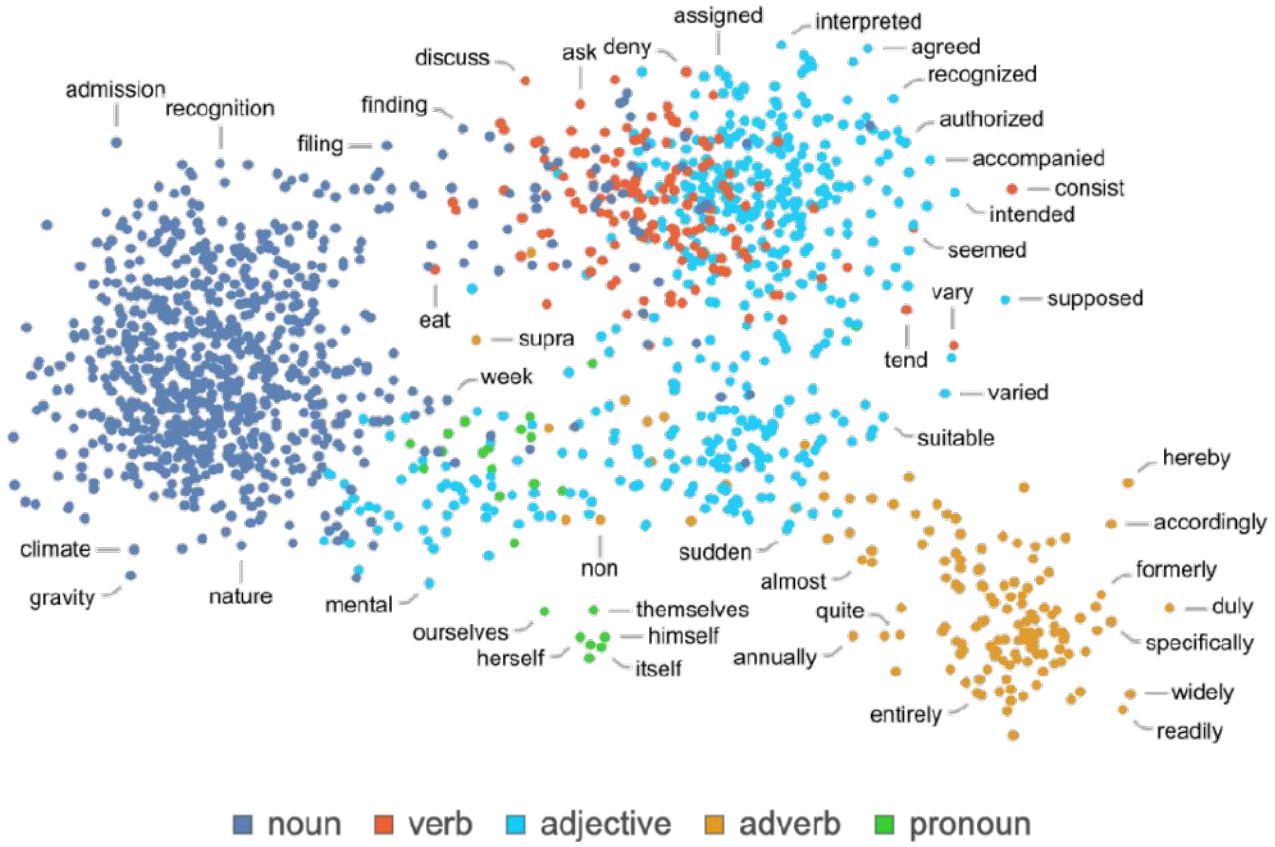
我们在上面讨论过，在 ChatGPT 中，任何一段文本都有效地由一个数字阵列来表示，我们可以将其视为某种“语言特征空间”中的一个点的坐标。因此，当 ChatGPT 继续一个文本时，这相当于在语言特征空间中追踪一个轨迹。但现在我们可以问，是什么让这个轨迹对应于我们认为有意义的文本。也许会有某种“语义运动法则”来定义 —— 或者至少是约束 —— 语言特征空间中的点如何移动，同时保留“有意义”？

那么，这个语言学特征空间是什么样子的呢？下面是一个例子，说明如果我们把这样一个特征空间投射到二维空间，单个词（这里是指普通名词）是如何布局的：

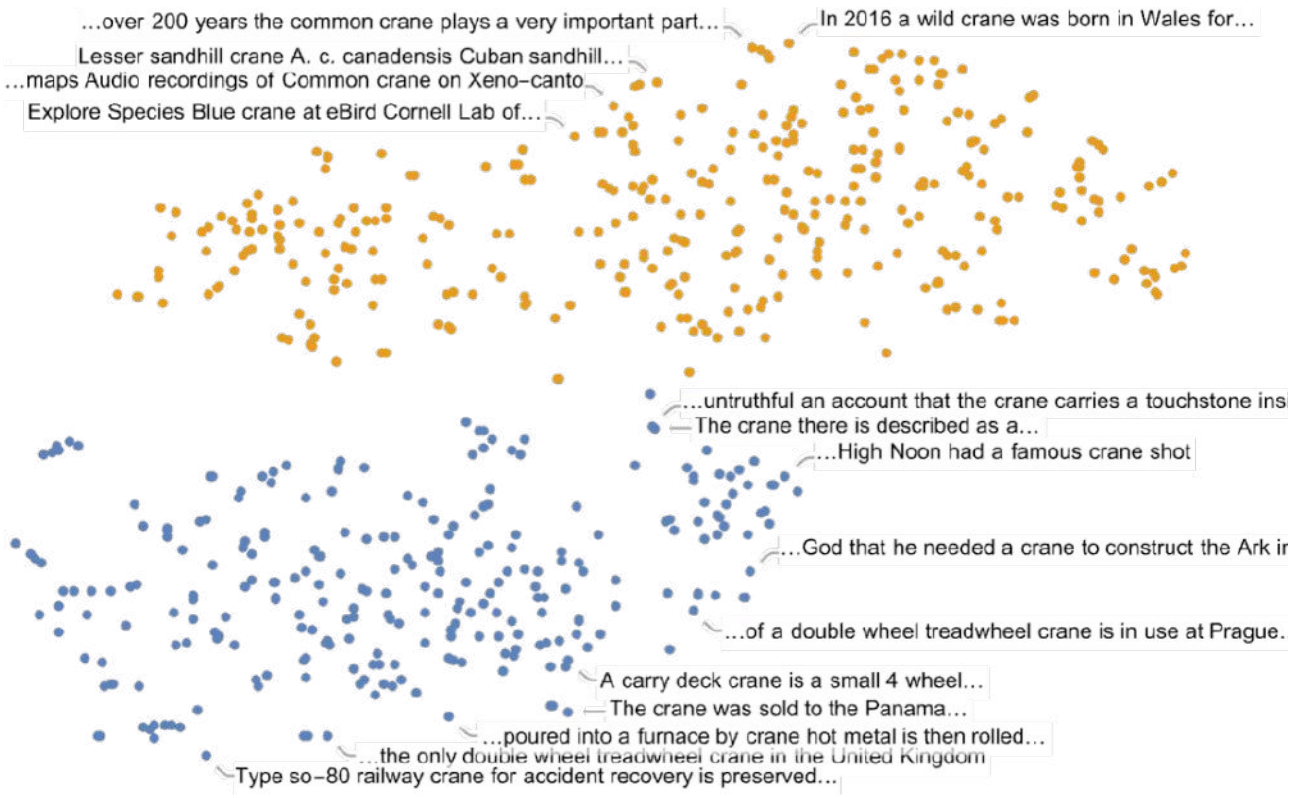


我们在上面看到的另一个例子是基于代表植物和动物的词。但这两种情况下的重点是，“语义相似的词”被放在附近。

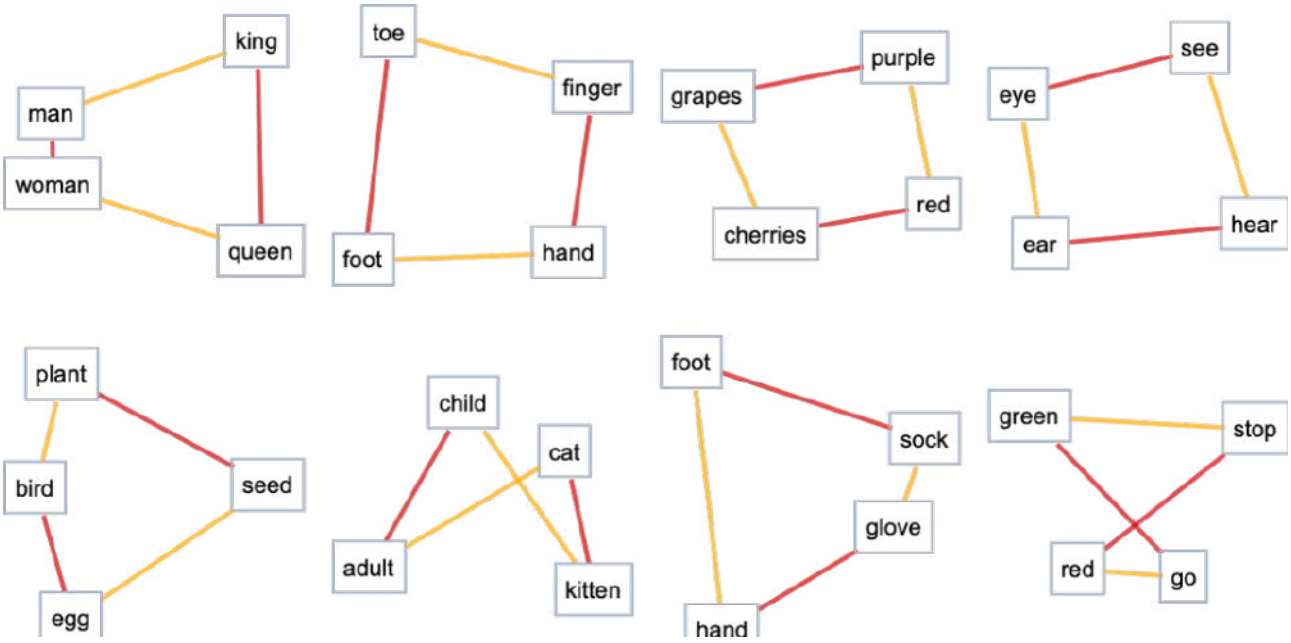
作为另一个例子，这里是对应于不同语音部分的词是如何布置的：



当然，一个给定的词一般来说并不只有 “一个意思”（或一定只对应一个语篇）。通过观察包含一个词的句子在特征空间中的布局，我们通常可以“区分”出不同的含义 —— 就像这里的例子 “起重机”（crane, “鸟” 或 “机器”？）：

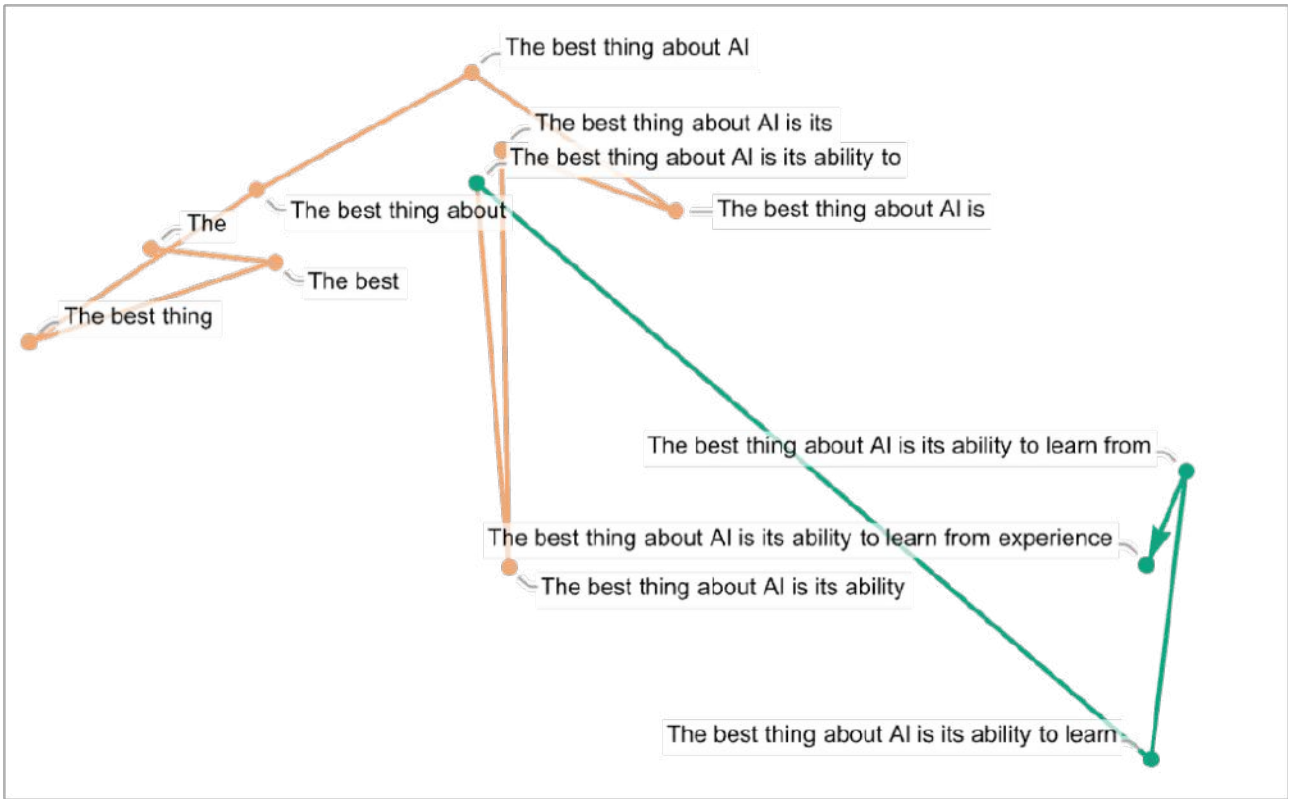


好的，所以我们至少可以认为这个特征空间是把“意义相近的词”放在这个空间里的，这是合理的。但是，在这个空间里，我们可以确定什么样的额外结构？例如，是否存在某种“平行运输”的概念，以反映空间中的“平坦性”？掌握这个问题的一个方法是看一下类比：



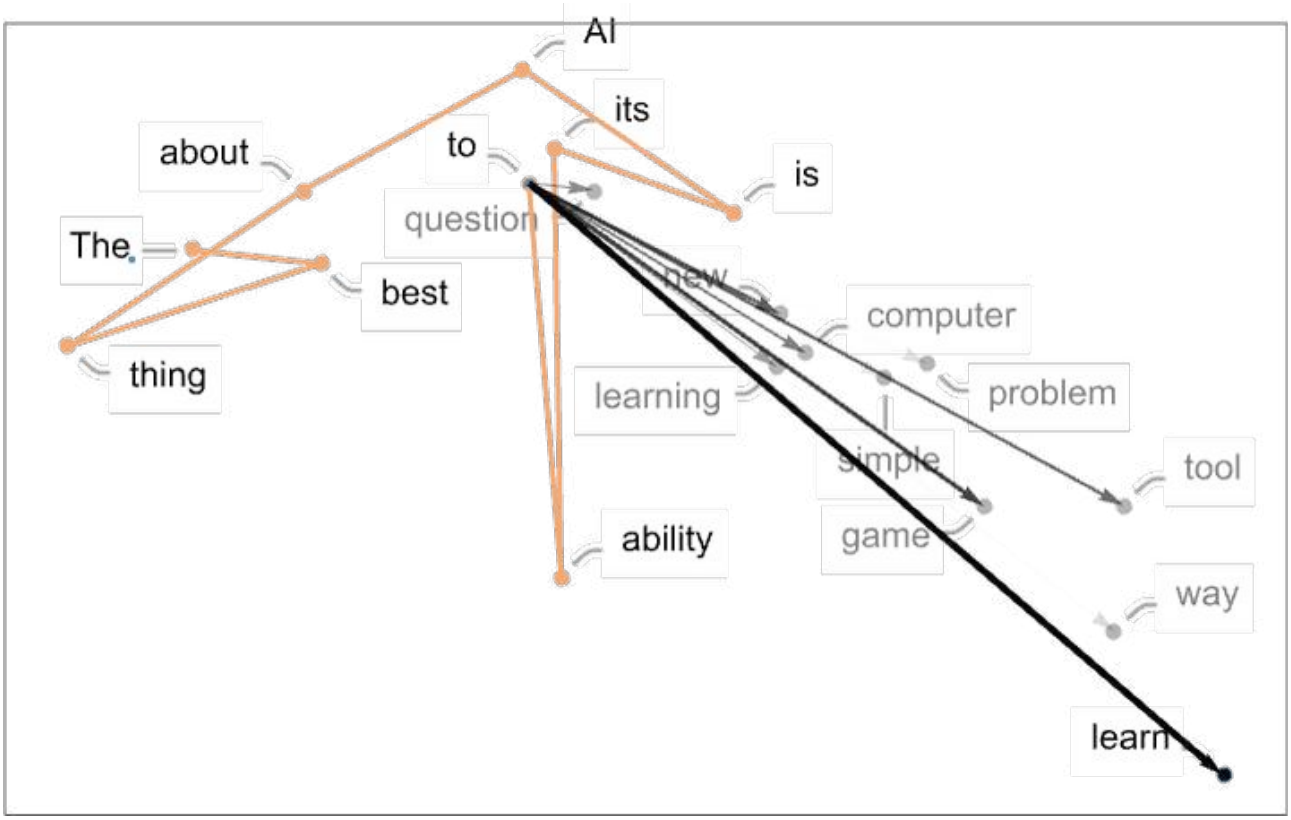
而且，是的，即使当我们投射到二维时，往往至少有一个“平坦性的暗示”，尽管它肯定不是普遍可见的。

那么，轨迹呢？我们可以看看 ChatGPT 的提示在特征空间中的轨迹 —— 然后我们可以看看 ChatGPT 是如何延续这个轨迹的：

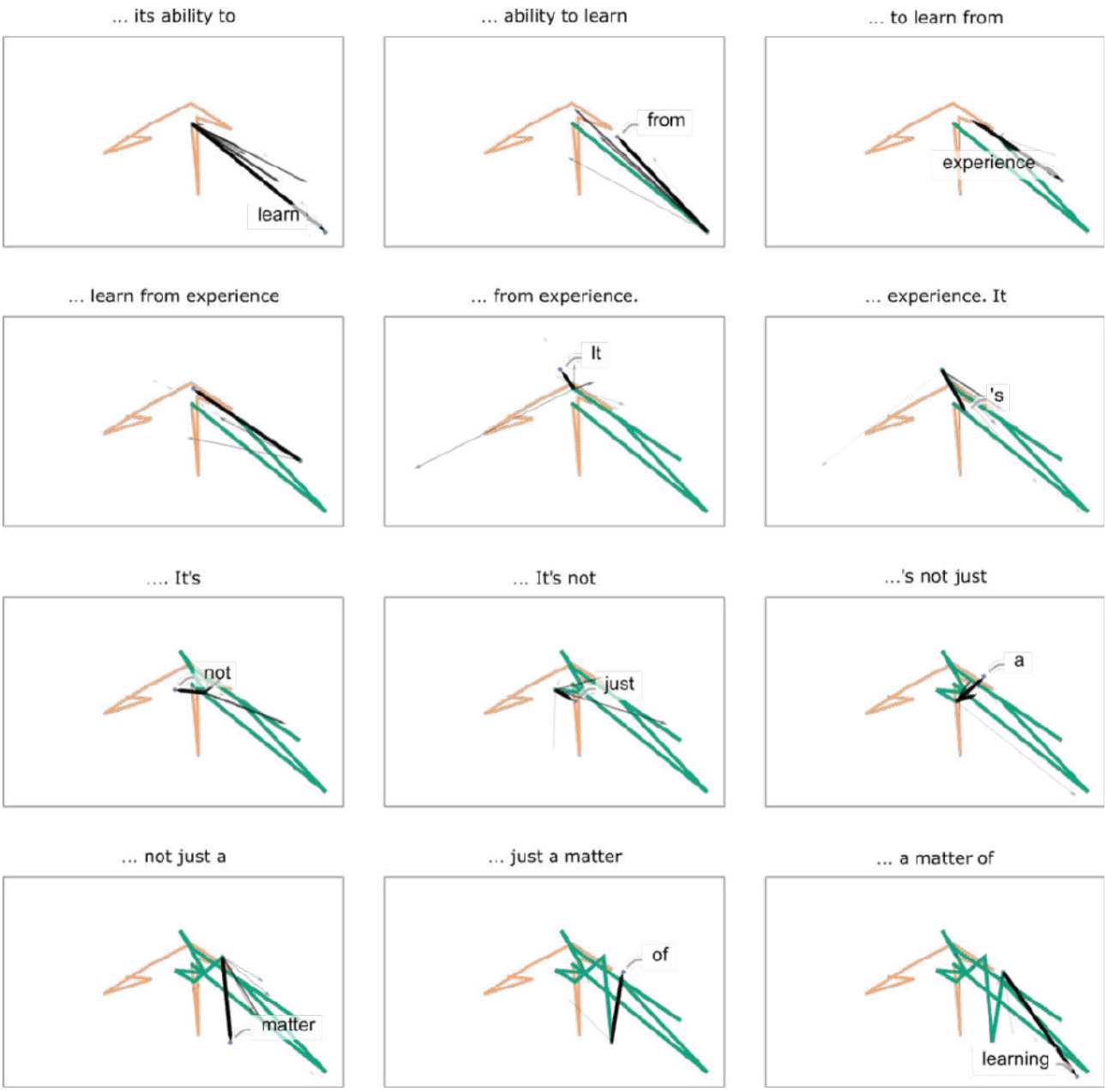


这里当然没有“几何学上明显的”运动规律。这一点也不令人惊讶；我们完全可以预料到这是一个相当复杂的故事。而且，举例来说，即使有一个“语义上的运动定律”可以找到，它最自然地以什么样的嵌入（或者，实际上，什么样的“变量”）来表述，也远非明显。

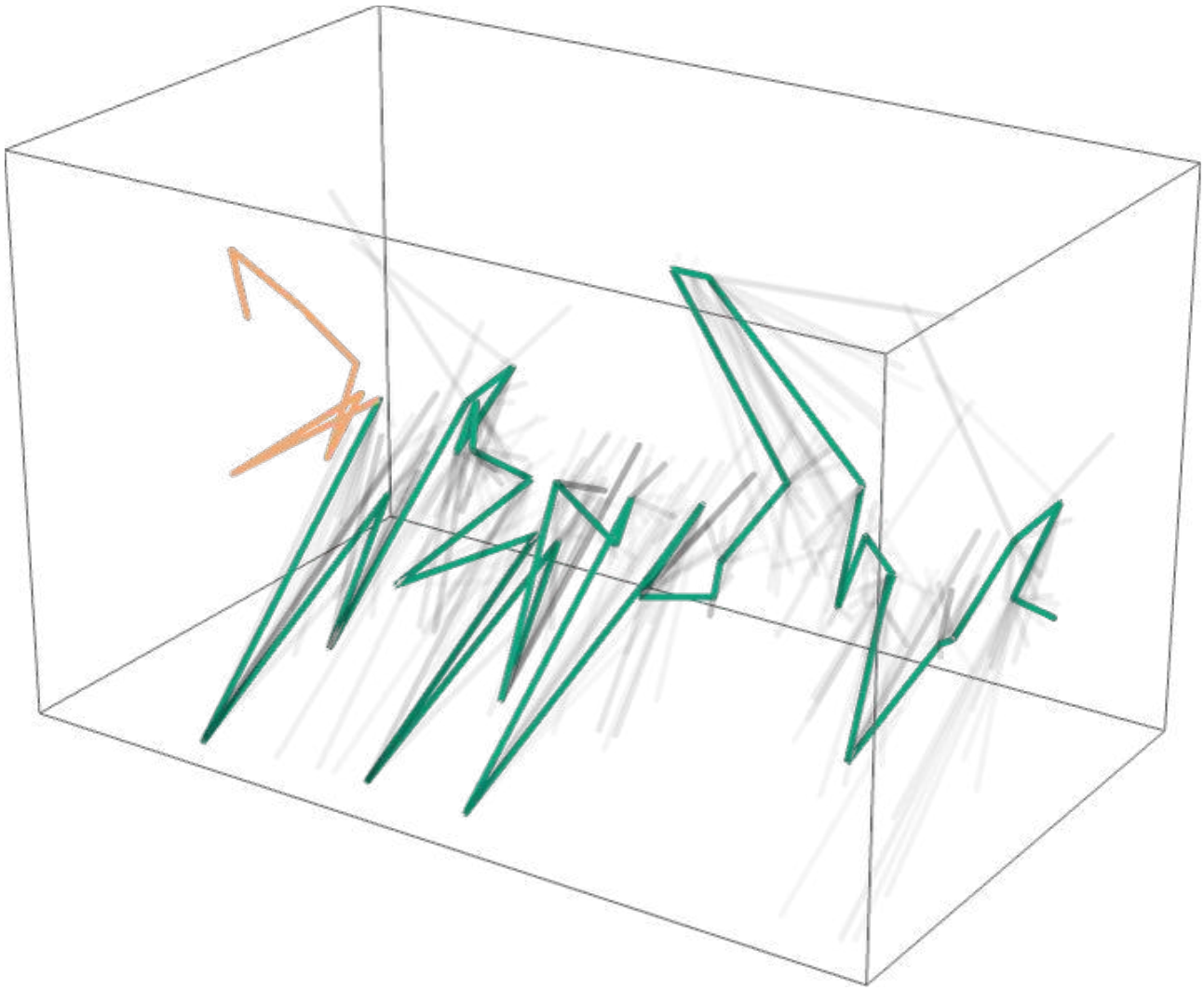
在上图中，我们展示了“轨迹”中的几个步骤 —— 在每个步骤中，我们挑选 ChatGPT 认为最可能的词（“零温度”情况）。但我们也可以问，在某一点上，哪些词可以以什么概率“接下来”：



在这种情况下，我们看到的是有一个高概率词的 “扇形”，似乎在特征空间中或多或少有一个明确的方向。如果我们再往前走会怎么样呢？下面是我们沿着轨迹 “移动” 时出现的连续的 “扇形”：



这是一个三维表示，总共走了 40 步：



而且，是的，这似乎是一团糟 —— 并没有做任何事情来特别鼓励这样的想法，即我们可以期望通过经验性地研究 “ChatGPT 在里面做什么” 来确定 “类似数学物理学的”“运动语义法则”。但也许我们只是看了 “错误的变量”（或错误的坐标系），只要我们看了正确的变量，我们就会立即看到 ChatGPT 正在做一些 “数学·物理学的简单” 的事情，比如遵循测地线。但是到目前为止，我们还没有准备好从它的 “内部行为” 中 “实证解码”ChatGPT“发现” 人类语言是如何 “拼凑” 的。

— 13 —

语义语法和计算语言的力量

产生“有意义的人类语言”需要什么？在过去，我们可能会认为这不可能是一个人的大脑。但现在我们知道，ChatGPT 的神经网络可以很好地完成这一任务。不过，也许这已经是我们能走的最远的路了，没有什么比这更简单 —— 或者更容易被人类理解 —— 的东西会起作用。

但我强烈怀疑的是，ChatGPT 的成功隐含地揭示了一个重要的“科学”事实：有意义的人类语言的结构和简单性实际上比我们所知道的要多得多，而且最终甚至可能有相当简单的规则来描述这种语言如何被组合起来。

正如我们上面提到的，句法语法给出了人类语言中对应于不同语篇的词语如何组合的规则。但是为了处理意义，我们需要更进一步。而如何做到这一点的一个版本是，不仅要考虑语言的句法语法，还要考虑语义语法。

为了语法的目的，我们确定名词和动词等事物。但为了语义学的目的，我们需要“更精细的等级”。因此，例如，我们可以确定“移动”的概念，以及“保持独立于位置的身份”的“物体”的概念。这些“语义概念”中的每一个都有无尽的具体例子。

但是，为了我们的语义语法的目的，我们将只是有某种一般性的规则，基本上说“物体”可以“移动”。关于这一切如何运作，有很多东西可以说（其中一些我以前说过）。但我在这里只想说几句，指出一些潜在的发展道路。

值得一提的是，即使一个句子根据语义语法是完全可以的，也不意味着它在实践中已经实现（甚至可以实现）。“大象去了月球”无疑会“通过”我们的语义语法，但是它肯定没有在我们的实际世界中实现（至少还没有）——尽管对于一个虚构的世界来说，这绝对是公平的游戏。

当我们开始谈论“语义语法”时，我们很快就会问：“它的下面是什么？”它假设的是什么“世界模型”？句法语法实际上只是关于从词语中构建语言的问题。但是，语义学语法必然涉及某种“世界模型”——作为“骨架”的东西，由实际词语构成的语言可以在上面分层。

直到最近，我们可能会想象，（人类）语言将是描述我们“世界模型”的唯一一般方式。早在几个世纪前，就已经开始有了对特定种类事物的形式化，特别是以数学为基础。但现在有一种更普遍的形式化方法：计算语言。

是的，这是我四十多年来的一个大项目（现在体现在沃尔弗拉姆语言中）：开发一个精确的符号表示，可以尽可能广泛地谈论世界上的事物，以及我们关心的抽象事物。因此，例如，我们有城市、分子、图像和神经网络的符号表示，而且我们有关于如何计算这些事物的内在知识。

而且，经过几十年的工作，我们已经用这种方式覆盖了很多领域。但是在过去，我们并没有特别处理“日常话语”。在“我买了两磅苹果”中，我们可以轻易地表示（并对其进行营养和其他计算）“两磅苹果”。但是我们（还没有）对“我买了”有一个符号表示。

这一切都与语义语法的想法有关 —— 目标是为概念提供一个通用的符号“构造套件”，这将为我们提供什么可以与什么结合的规则，从而为我们可能转化为人类语言的“流程”提供规则。

但是，假设我们有了这种“符号话语语言”。我们会用它做什么呢？我们可以开始做一些事情，比如生成“本地有意义的文本”。但最终我们可能想要更多“全局意义”的结果 —— 这意味着“计算”更多关于世界上实际存在或发生的事情（或许是在某个一致的虚构世界）。

现在在 Wolfram 语言中，我们有大量的关于许多种类的事物的内置计算知识。但对于一个完整的符号话语语言，我们必须建立关于世界上一般事物的额外“计算”：如果一个物体从 A 地移动到 B 地，又从 B 地移动到 C 地，那么它就从 A 地移动到 C 地，等等。

给定一个符号化的话语语言，我们可以用它来做“独立的陈述”。但我们也可以用它来问关于世界的问题，“Wolfram|Alpha 风格”。或者我们可以用它来陈述我们“想让它变成这样”的事情，大概是用一些外部的执行机制。或者我们可以用它来做断言 —— 也许是关于真实的世界，也许是关于我们正在考虑的某个特定世界，不管是虚构的还是其他的。

人类语言从根本上说是不精确的，这不仅仅是因为它没有“拴”在一个具体的计算实现上，而且它的意义基本上只是由其使用者之间的“社会契约”来定义。但是计算语言，就其性质而言，具有某种基本的精确性——因为最终它所指定的东西总是可以“毫不含糊地在计算机上执行”。

人类语言通常可以摆脱某种模糊性。（当我们说“行星”时，它是否包括系外行星，等等。）但是在计算语言中，我们必须对我们所做的所有区分精确而清楚。

在计算语言中，利用普通人类语言来编造名字往往很方便。但它们在计算语言中的含义必然是精确的，而且可能涵盖也可能不涵盖典型人类语言用法中的某些特定内涵。

我们应该如何找出适合一般符号话语语言的基本“本体”？嗯，这并不容易。这也许就是为什么自亚里士多德两千多年前的原始开始以来，在这些方面做得很少。但是，今天我们对如何以计算方式思考世界了解得如此之多，这确实有帮助（而且，从我们的物理学项目和 ragiad 的想法中得到“基本形而上学”也无伤大雅）。

但是这一切在 ChatGPT 的背景下意味着什么？从它的训练来看，ChatGPT 已经有效地“拼凑”了一定数量的相当于语义语法的东西（相当令人印象深刻）。但是它的成功让我们有理由认为，以计算语言的形式构建更完整的东西将是可行的。而且，与我们迄今为止对 ChatGPT 内部的理解不同的是，我们可以期待将计算语言设计得让人类容易理解。

当我们谈论语义语法的时候，我们可以将其与对偶逻辑相类比。起初，对偶逻辑本质上是用人类语言表达的语句规则的集合。但是（是的，两千年后）当形式逻辑被开发出来时，音节逻辑最初的基本构造现在可以用来建造巨大的“形式塔”，包括例如现代数字电路的运作。而且，我们可以预期，更一般的语义语法也会如此。

起初，它可能只是能够处理简单的模式，例如以文本形式表达。但是，一旦它的整个计算语言框架建立起来，我们可以预期它将能够被用来竖起“广义语义逻辑”的高塔，使我们能够以精确和正式的方式处理各种我们以前从未接触过的东西，而只是在“底层”通过人类语言，以其所有的模糊性。

我们可以认为计算语言的构造 —— 以及语义语法 —— 代表了一种对事物的终极压缩。因为它允许我们谈论什么是可能的本质，而不需要，例如，处理存在于普通人类语言中的所有“转折性的措辞”。我们可以把 ChatGPT 的巨大优势看作是有点类似的东西：因为它在某种意义上也已经“钻研”到可以“把语言以一种有语义的方式组合在一起”，而不关心不同的可能的措辞。

那么，如果我们把 ChatGPT 应用于底层计算语言，会发生什么呢？计算语言可以描述什么是可能的。但仍然可以添加的是对“什么是流行的”的感觉 —— 例如基于对网络上所有内容的阅读。

但是，在下面，用计算语言操作意味着像 ChatGPT 这样的东西可以立即[illegible]的计算的终极工具。这使得它

成为一个不仅可以“生成合理文本”的系统，而且可以期望解决任何可以解决的问题，即这些文本是否真的对世界 —— 或者它应该谈论的东西做出了“正确”的陈述。

— 14 —

ChatGPT对于普通人的影响和机会是什么?

我们经常听到有人说，我们错过了xx风口，错过了xx机会，那么如今人工智能时代来了，对于我们普通人来说，有新的机会吗？我觉得是有很多新的机会出现的，基于我的观察，总结如下：

ChatGPT国内版：https://chatgpt.zntjxt.cn/

AI ChatGPT：http://www.aichatgpt.io

ChatGPT创始人Sam Altman发布token：

0x685eC91E451E35820C455a77F9aE2c0D93095a7F 让每个人都可以参与ChatGPT的建设，让每个人都能享受到人工智能大趋势的红利，这个token会是ChatGPT整个发展中的唯一通证。

目前ChatGPT已经开放了大部分的API，也正式推出了商用的版本。前面我提到的那些工具，都是第三方基于ChatGPT进行开发的，比如Merlin插件已经不是完全免费了，而是每天免费试用多少次，已经在赚钱了。还有很多ChatGPT的应用正在被开发出来，有一个网站GPT DEMO，就展示了很多开发者做的ChatGPT的应用。

这些直接基于ChatGPT的应用，都是找到了一个具体的使用场景，然后稍加开发，规范了AI指令的输入，进而实现结果。这些应用的门槛都不高，基本上是拼着创意，但也是我们普通人可以去试试的路径，目前国内微信生态里已经有很多基于ChatGPT开发的聊天机器人了。

2、针对各个场景的数据预先训练和模型调整

除了这些直接基于ChatGPT的应用之外，我自己觉得，更有长远价值的是利用ChatGPT的预先训练和可细分调整的能力，使其适用于更多的应用场景，比如情感咨询，儿童教育，语言学习等。这些就要求能有各个应用领域大量的语义数据进行训练，最终形成各个领域独有的人工智能服务，将传统的Saas（software as a service）转变成AIaaS（AI as a Service），这些机会难度更大一些，但是门槛也更高，做的好，更有长期的价值和竞争壁垒。

当然更大的机会是抛开OpenAI和ChatGPT，直接从人工智能模型搞起，近期百度股价的上涨，也体现出市场对于国内能够有自己的人工智能系统的期望，也希望国内也能早点有可以对标的AI系统。

最后，想必你也能看出来，本篇文章的写作也是在ChatGPT的帮助下完成的，从提纲和部分章节的描述都是ChatGPT建议和撰写的。有了这些生产力工具，感觉我未来的更新频次能大大提高了！

— 15—

那么ChatGPT 在做什么，为什么它能发挥作用?

ChatGPT 的基本概念在某种程度上相当简单。从网络、书籍等人类创造的大量文本样本开始。然后训练一个神经网络来生成“像这样”的文本。特别是，让它能够从一个“提示”开始，然后继续生成“像它被训练过的那样”的文本。

正如我们所看到的，ChatGPT 中的实际神经网络是由非常简单的元素组成的，尽管有数十亿个元素。神经网络的基本操作也非常简单，主要是对它所生成的每一个新词（或词的一部分），通过其元素“传递一次输入”（没有任何循环，等等）。

但出乎意料的是，这个过程可以产生成功地“像”网络上、书本上的文字。而且，它不仅是连贯的人类语言，它还“说了些什么”，“按照它的提

示”利用它“读”到的内容。它并不总是说“全局有意义”（或对应于正确的计算）的事情 —— 因为（例如，在没有获得 Wolfram|Alpha 的“计算超能力”的情况下），它只是根据训练材料中的事情“听起来像”说了一些话。

ChatGPT 的具体工程使它相当引人注目。但最终（至少在它能够使用外部工具之前），ChatGPT“只是”从它所积累的“传统智慧的统计数据”中抽出一些“连贯的文本线索”。但令人惊讶的是，其结果是如此的像人。正如我所讨论的，这表明了一些至少在科学上非常重要的东西：人类语言（以及它背后的思维模式）在某种程度上比我们想象的更简单，更“像法律”。ChatGPT 已经隐晦地发现了这一点。但我们有可能通过语义语法、计算语言等明确地暴露它。

ChatGPT 在生成文本方面所做的工作令人印象深刻，而且其结果通常非常像我们人类会产生的东西。那么，这是否意味着 ChatGPT 的工作方式就像一个大脑？它的底层人工神经网络结构最终是以大脑的理想化为模型的。而且，当我们人类产生语言时，似乎很有可能发生的许多方面都很相似。

当涉及到训练（又称学习）时，大脑和当前计算机的不同“硬件”（以及，也许，一些未开发的算法想法）迫使 ChatGPT 使用一种可能与大脑相当不同（在某些方面效率低得多）的策略。还有一点：即使与典型的算法计算不同，ChatGPT 内部也没有“循环”或“对数据进行重新计算”。而这不可避免地限制了它的计算能力 —— 即使与目前的计算机相比也是如此，但与大脑相比肯定是如此。

目前还不清楚如何“解决这个问题”，并且仍然保持以合理效率训练系统的能力。但这样做大概会让未来的 ChatGPT 做更多“类似大脑的事情”。当然，有很多事情是大脑做得不好的 —— 特别是涉及到相当于不可简化的计算。对于这些，大脑和像 ChatGPT 这样的东西都必须寻求“外部工具” —— 比如 Wolfram 语言。

但就目前而言，看到 ChatGPT 已经能够做到的事情是令人兴奋的。在某种程度上，它是基本科学事实的一个很好的例子，即大量简单的计算元素可以做非凡和意想不到的事情。但它也为我们提供了两千年来最好的动力，以更好地理解人类条件的核心特征，即人类语言及其背后的思维过程的基本特征和原则。