章节修订历史

本表格记录修订版本之间的重大改动。类似简单说明或者变更格式这样的细微修改并不会加以记录。

版本号	改动日期	改动内容	负责人
0.4		Initial release	A. Lavin, S. Ahmad, J. Hawkins
0.41	Dec 21, 2016	Replaced figure 5 and made some clarifications.	S. Ahmad
0.42	May 30, 2017	Correction to equation (12)	S. Ahmad

稀疏分布表征

本章我们介绍稀疏分布表征（SDR），这是脑和 HTM 系统中的信息表征的基本形式。我们会讨论 SDR 的数个既有趣又有用的数学特征，并且论述在脑中 SDR 是如何运用的。

什么是稀疏分布表征？

AI 领域最有趣的挑战之一是知识表征问题。采取计算机可以处理的某种形式来表示日常事实和关系已经被证实是很难通过传统计算机科学的方法完成的。最根本的问题在于我们对世界的认识没有划分成具有明确关系的离散事实。几乎所有我们知道的事情都有例外，而且概念之间的关系太多、不明确，所以无法映射到传统的计算机数据结构。

脑没有这个问题。它们运用一种称作稀疏分布表征的方法，即 SDR 来表示信息。SDR 以及它的数学特征对生物智能至关重要。脑做的每件事、本书描述的每条原理，都是基于 SDR。SDR 是脑的语言。人类智能的灵活性和创造性离不开这种表征方法。所以，如果我们想让智能机器具有同样的灵活性和创造性，它们就要基于相同的表征方法，也就是 SDR。

一条 SDR 由数千个比特组成，在任何时刻，有一小部分的比特是置 1 的，其余的是置 0 的。SDR 中的比特对应于脑中的神经元，置 1 的是相对激活的神经元，置 0 的是相对抑制的神经元。SDR 最重要的特征是其中每个比特都有意义。所以，任何特定表征中的激活比特编码了所表示的语义属性。这些比特没有被标记（也就是说，比特没有被指定其含义），与之相反，比特的语义是习得的。如果两条 SDR 在相同的位置上有激活比特，那么它们共享由这些比特表示的语义属性。通过确定两条 SDR 的重叠部分（两条 SDR 中对应位置都置 1 的比特），我们可以很快理解两个表征是如何在语义上相似，以及它们在如何在语义上不同。由于这种语义重叠特征，基于 SDR 的系统能够自然而然地以语义相似性为基础推而广之。

HTM 理论规定了如何创建、存储、回忆 SDR 和 SDR 序列。SDR 不像计算机的数据那样在内存中来来去去，反而像在固定的神经元群中的激活神经元，随着时间而变化。在某个时刻，一组神经元代表某个意思；在下个时刻，它就代表其他的意思。在一组神经元中，在某个时刻的 SDR 可以关联地链接紧接着要出现的 SDR。这样，SDR 的序列就被学习到了。关联链接也出现在不同的细胞群之间（层次之间或者区域之间）。一个区域的神经元编码的含义不同于另一个区域的神经元编码的含义。这样，一种形态的 SDR，比如声音，可以关联调用另一种形态的 SDR，比如视觉。

任何类型的概念都可以编码到一条 SDR 中，包括不同类型的传感器数据，比如词语、位置和行为。这也是为什么新皮质是通用的学习机器。新皮质的各个区域操作 SDR，并不用"知道"它们在现实世界代表什么。HTM 系统的运作方式是相同的。只要输入信息采用了适当的 SDR 格式，HTM 算法就可以工作。在基于 HTM 理论的系统中，知识是数据所固有的，而不是算法中的。

为了更好地理解 SDR 的特征，回想一下计算机中的信息普遍是如何表示的，以及 SDR 相对于计算机的信息表征体系的优缺点，这样会有帮助。在计算机中，我们用比特和字来表示信息。比如，要表示内科病人的信息，计算机程序可能会用一个字节存储病人的年龄，用另一个字节存储病人的性别。用列表或者树这样的数据结构组织相关的信息。如果我们需要表示的信息定义良好并且程度有限，这种表征类型就很有效。但是，AI 研究员认识到要让计算机变得智能，就需要它接触大量的知识，而这些知识的结构没有明确定义。

打个比方，如果我们想要我们的智能计算机了解汽车该怎么办？想想所有你知道的关于汽车的事情。你知道它们的用途、怎么驾驶它们、如何进去出来、怎么清理它们、可能会出现的故障、不同的控制键的功能、引擎盖下面的东西、如何更换轮胎等等。我们知道汽车的形状和它们发出的声音。如果你只考虑轮胎，你可能会想起不同类型的轮胎、不同

品牌的轮胎、它们如何不均匀地磨损、转动它们的最佳方式等等。记录你所知道的关于汽车的事情的清单会像这样继续下去。

每一条知识都会在不断拓展的关联网络中引出其他知识。比如，汽车有车门，但是其他东西也有门，像房屋、飞机、邮筒和电梯。我们可以凭着直觉知道所有这些门的相似和不同之处，还可以根据以往的经验对我们从未见过的新型门进行预估。我们（我们的脑）容易通过关联来回忆大量事实和关系。当 AI 学者试图把这种知识编码到计算机中时，他们遇到了困难。

在计算机中，信息是用 8、32 或者 64 位比特的字表示的。置 1 和置 0 的所有组合都用上了，从全部置 1 到全部置 0，这有时称作一种“稠密”表征。稠密表征的一个例子是字母表中字母的 ASCII 码。ASCII 码中的字母“m”表示为：

01101101

注意，全部八个比特的这个组合编码了“m”，其中单独的比特自身不表示任何意思。你说不出第三个比特的意思，全部八个比特的组合是必要的。也要注意稠密表征是脆弱的。比如，如果你像这样改变“m”的 ASCII 码中的一个比特：

01100101

你得到了一个完全不同的字母，“e”的表征。一个错误的比特导致意思完全变了。

像 01101101 这种稠密表征不能告诉你它表示什么或者它有什么属性。它完全是一种主观、抽象的表征。稠密表征的含义必须存储在其他某个地方。

相比之下，正如之前提到的，SDR 有数千个比特。在任何时刻，有一小部分的比特是置 1 的，其余的是置 0 的。一条 SDR 可以有 2000 个比特，其中 2%（即 40 个）是置 1 的，其余的置 0。如图 1 所示。

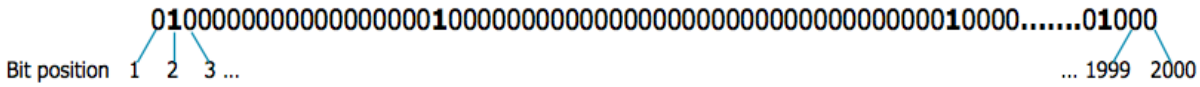

图 1 一条有 2000 个比特的 SDR，其中仅有少数是 ON（置 1 的）。

在一条 SDR 中，每个比特都有意义。比如，如果我们想用 SDR 表示字母表中的字母，可能会有比特用来表示字母是辅音还是元音、字母的发音如何、字母出现在字母表中的位置、字母怎样书写（也就是形状是开还是闭、是上行还是下行）等等。为了表示一个特定的字母，我们选出能够最好地描述字母的 40 个属性，并把对应的比特置 1。事实上，每个比特的含义是习得的而非指定的。我们用字母和它们的属性来讲解原理。

SDR 通过一组激活比特来表示东西的含义。所以，如果两个不同表征在相同位置上的比特都置 1，我们可以确定这两个表征共享了这个属性。因为表征是稀疏的，两个表征不会偶然地共享某个属性；共享的比特 / 属性总是有意义的。如图 2 所示，用这种方式简单地比较 SDR 可以知道两个对象在语义上是怎样地相似和不同。

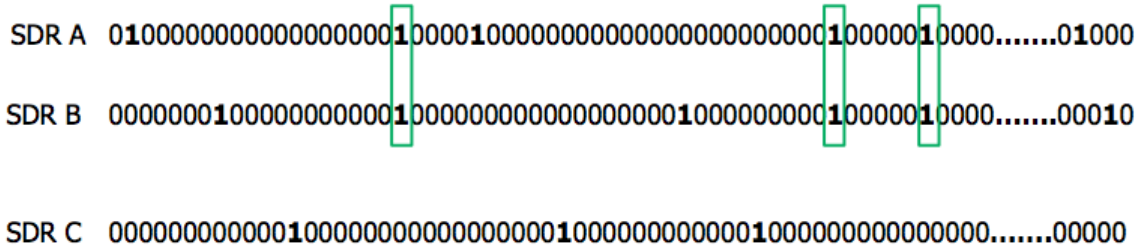

图 2 SDR A 和 SDR B 有匹配的置 1 比特，所以共享对应的语义；SDR C 没有匹配的置 1 比特，也就没有共享的语义。

SDR 的一些不可思议的数学特征是计算机的传统数据结构所不具备的。比如，要把一条 SDR 存储到内存中，我们不需要像处理稠密表征那样存储所有比特。我们只要存储置 1 比特的位置，而且惊人的是，我们只要存储少量的置 1 比特的位置。假设我们有一条 10,000 个比特大小的 SDR，其中的 2%也就是 200 个比特是置 1 的。为了存储这条 SDR，我们记住这 200 个置 1 比特的位置。要和一条新的 SDR 做比较，我们看看新的 SDR 中的这 200 个位置的比特是否都是置 1 的。如果是，那么和新的 SDR 就是一样的。现在想象一下，如果我们只存储从原先 200 个比特中随机选出的 10 个比特的位置。要比较两条 SDR 是否一样，我们看新的 SDR 的这 10 个位置的比特是否是置 1 比特。你可能会想，“但是，等一下，在很多种情况下，这 10 个比特是匹配的，但是其他比特不匹配。我们可能很容易碰上误报匹配！”。的确如此。但是，随机挑选的 SDR 共享 10 个比特的概率非常低；这种情况不会偶然发生，所以存储这 10 个比特就够了。但是，如果两条 SDR 确实都在这 10 个相同的位置上有置 1 比特，那么这两个 SDR 在语义上是相似的。用同样的方式处理它们是一种有用的泛化形式。我们会在本章的后面论述这个有趣的特征，并且对它进行数学推导。

SDR 的另一个不可思议并且有用的特征是合并特征，如图 3 所示。我们可以用一组 SDR 形成一条新的 SDR，这是原先一组 SDR 的合并。要形成一个合并 SDR，我们只要让所有的 SDR 共同做逻辑 OR 操作。得到的合并 SDR 和原先的每条 SDR 都有一样数目的比特，但是它不那么稀疏。形成合并是一种单向操作，就是说你无法根据形成的合并 SDR 得知是用哪些 SDR 来形成它的。但是你可以用一条新的 SDR 和这个合并 SDR 比较，来确定新的 SDR 是否是参与合并的某条 SDR。因为 SDR 的稀疏性，所以判断失误的概率很低。

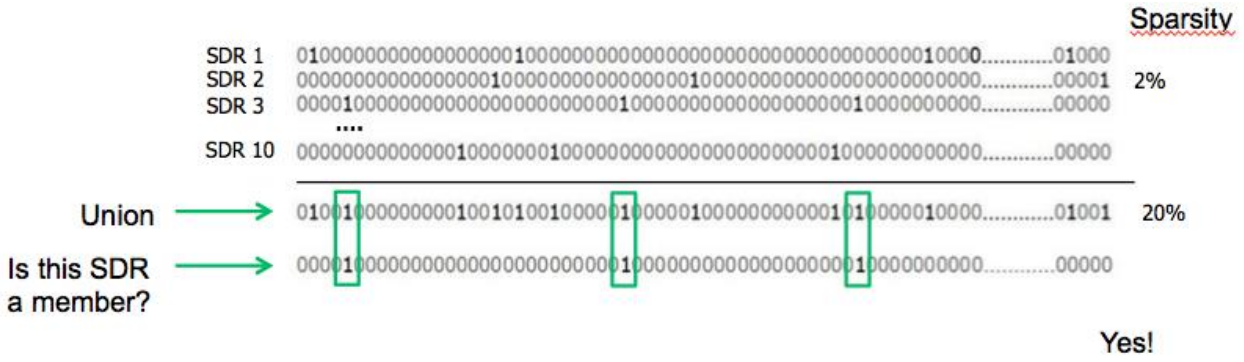

图3 这10条SDR的合并是对比特序列做逻辑OR操作而形成的。新的SDR是否属于那些原先的SDR是通过检查它与合并SDR匹配的置1比特来确定的。注意，合并SDR不如原先的那些SDR稀疏。

这些特征，以及一些其他的，在实践中非常有用，触及到了人脑之所以不同于计算机的根本原因。接下来的几节会更详细地描述这些特征以及 SDR 的操作。在本章的最后，我们会论述 SDR 在人脑和 HTM 中的一些运用方式。

SDR 的数学特征

本节我们论述以下的 SDR 的数学特征，着重推导根本的标度律和误差界：

- SDR 的容量和错误匹配的概率
- SDR 的健壮性和噪声误差的概率
- SDR 向量的可靠分类
- SDR 的合并
- SDR 的合并在噪声环境下的健壮性

这些特征和相关的操作证实了 SDR 作为存储空间的实用性，我们会在与 HTM 相关的例子中说明。在我们的分析中，我们依赖由 Kanerva 提供的关于直觉的研究成果【Kanerva, 1988 & 1997】和一些用于分析布隆过滤器的技术【Bloom, 1970】。我们会以相应的概述开始每个特征的论述，然后进行特征所涉及到的数学推导。但是首先，这里有一些术语的定义和符号，我们会在接下来的论述和整本书中用到。更详尽的关于术语的清单可以在本书的末尾找到。

数学定义和符号

二进制向量：为了方便讨论，我们把 SDR 看作是**二进制向量**，用 $\mathbf{x} = [b_0, \ldots, b_{n-1}]$ 作为 SDR $\mathbf{x}$ 的定义。每个元素的值是“0”或者“1”，分别表示 OFF 和 ON。

向量模：在 SDR $\mathbf{x} = [b_0, \ldots, b_{n-1}]$ 中，n 指代**向量的长度**。也就是说，我们用 n 表示向量中元素的总数、向量的维数或者比特的总数。

稀疏度：在任何时刻，向量 $\mathbf{x}$ 中的 n 个比特是 ON，其余的是 OFF。用 s 指代 ON 比特所占的比例。一般在稀疏表征中，s 要远远低于 50%。

向量势：用 w指代**向量的基数**，我们定义为向量中 ON 比特的总数。如果向量 $\mathbf{x}$中的 ON 比特所占比重是 s，那么有 $w_{\mathbf{x}} = s \times n = \|\mathbf{x}\|_0$。

重叠：我们通过**重叠评分**来确定两条 SDR 的相似程度。简单地说，重叠评分就是共有的 ON 比特的数目，也就是向量中相同位置的 ON 比特的数目。如果 $\mathbf{x}$ 和 $\mathbf{y}$ 是两条 SDR，那么重叠评分可以用点乘运算得到：

$$overlap(\mathbf{x}, \mathbf{y}) \equiv \mathbf{x} \cdot \mathbf{y}$$

注意，我们没有采用典型的距离度量公式，比如汉明距离或者欧式距离，来量化相似程度。可以用重叠推导出一些有用的特征，我们会在后续进行论述，它们和距离度量方法是不相容的。

匹配：我们通过检查两条 SDR 是否充分重叠来确定**匹配**。对 $\mathbf{x}$ 和 $\mathbf{y}$ 这两条 SDR 而言：

$$match(\mathbf{x}, \mathbf{y}|\theta) \equiv overlap(\mathbf{x}, \mathbf{y}) \geq \theta$$

如果 $\mathbf{x}$ 和 $\mathbf{y}$ 有相同的势 w，我们可以通过设定阈值 $\theta = w$ 来确定一个精确匹配。这样的话，如果θ小于 w，重叠评分会表明这是一个**非精确匹配**。

假设有两个 SDR 向量：

$$\mathbf{x} = [01000000000000000001000000000000110000000]$$
$$\mathbf{y} = [10000000000000000001000000000000110000000]$$

两个向量都有相同的参数，向量模 $n = 40$，稀疏度 $s = 0.1$，向量势 $w = 4$。向量 x 和 y 的重叠评分是 3；也就是说，这两个向量中有三个处在相同位置的 ON 比特。因此当设定 $\theta = 3$ 时这两个向量是匹配的，但是它们并非一个精确匹配。注意，如果阈值大于其中任意向量的势，也就是 $\theta > w$，那么说明了任何匹配都不可能发生。

SDR 的容量和错误匹配的概率

要在实践中有用，SDR 应该有很大的容量。给定一个模为 n 且势为 w 的向量，那么对它编码的不同 SDR 的数目可以用从 n 个中选出 w 个的组合数表示：

$$\binom{n}{w} = \frac{n!}{w!\,(n-w)!} \tag{1}$$

注意，对于同样大小的向量，稠密表征可用的编码总数是 2^n，相比之下 SDR 的明显要更小。这意味着潜在的容量损失，因为输入模式可能出现的情况数目要比用 SDR 编码可用的表征数目大得多。尽管 SDR 的容量要比稠密编码小很多，实际上这没什么影响。如果选取典型值，比如模 $n = 2048$ 和势 $w = 40$，SDR 的表征空间就有天文数字那么大，可以有 2.37×10^{84} 个编码；而可观测宇宙中的原子数估计为 $\sim 10^{80}$。

要让SDR能够表示信息，我们需要可靠地区分出编码；也就是说，SDR应该相互区别，这样我们才不会混淆信息。只有这样，接下来理解两个随机 SDR 的相同概率才有意义。给定两个有着同样参数的随机 SDR x 和 y，它们相同的概率是：

$$P(x = y) = 1/\binom{n}{w} \tag{2}$$

假设一种情况，向量模为 $n = 1024$ 且向量势为 $w = 2$。会有 523776 种可能的编码，其中两个随机编码相同的可能性非常高，也就是 523,776 分之 1。这个概率会随着 w 的升高而非常迅速地降低。当 $w = 4$ 时，概率骤降到不足 450 亿分之 1。如果模 $n = 2048$ 且势 $w = 40$，这是 HTM 的典型值，两个随机编码相同的概率基本上是零。请注意等式 2 反映的是精确匹配条件下的误报概率，而非大多数 HTM 模型采用的非精确匹配；本章后面会对此进行论述。

上述的等式说明了，SDR 具备足够大的尺寸和密度时，会有超乎寻常的容量满足不同编码，不同的表征几乎不可能用上相同的编码。

重叠集

我们引入重叠集的概念来帮助分析在不同条件下的匹配效果。假设 x 是一个大小为 n 且有 w_x 个比特为 ON 的 SDR 编码。向量 x 关于新参数 b 的重叠集是 $\Omega_x(n, w, b)$，定义为大小为 n 且有 w 个比特为 ON，与向量 x 恰好有 b 个比特重叠的向量的集合。这些向量的数目用 $|\Omega_x(n, w, b)|$ 表示，这里的 $|\cdot|$ 表示集合中所有元素的数目。如果 $b \leq w_x$ 且 $b \leq w$，那么有：

$$|\Omega_x(n, w, b)| = \binom{w_x}{b} \times \binom{n - w_x}{w - b} \tag{3}$$

等式 3 中乘积的第一个因数表示在向量 x 中的 w_x 个 ON 比特对应的位置出现 b 个 ON 比特的组合数，第二个因数表示在向量 x 中的 $n - w_x$ 个 OFF 比特对应的位置出现 $w - b$ 个 ON 比特的组合数。其中前者体现的是重叠集中的向量与向量 x 重叠的部分。

重叠集很有启发性，我们可以通过它可靠地比较 SDR；也就是不产生漏报或者误报，甚至存在明显噪声时，这意味着 ON / OFF 比特会随机波动。在接下来的几节，我们会用非精确匹配和子采样两种不同的思想来探究 SDR 在噪声环境下的健壮性。

非精确匹配

如果我们要求两条 SDR 做精确匹配（也就是 $\theta = w$），那么只要任意两条中的 ON 比特有一个噪声比特，就会产生漏报，我们也就不能识别匹配的 SDR。一般而言，我们想要系统能够容忍输入信息中的变动或者噪声。这就是说，我们很少需要精确的匹配，也就是 $\theta = w$ 的情形。降低 θ 让我们可以使用非精确匹配，减小了系统灵敏度而且增大了对抗噪声的整体健壮性。比如，考虑 SDR 向量 x 和 x'，其中 x' 是随机噪声污染 x 后的值。设定 $w = 40$ 并且把 θ 降低到 20，噪声会翻转 50%的比特（ON 比特变成 OFF 比特，反之亦然），但是 x 仍然能匹配 x'。

但是增大健壮性伴随着的代价是更多的误报。就是说，减小θ也增大了与另一个随机向量发生错误匹配的概率。在调整这些参数时有一个固有的折衷。正如我们所希望的，在保持健壮性的同时，尽可能地降低发生错误匹配的可能性。

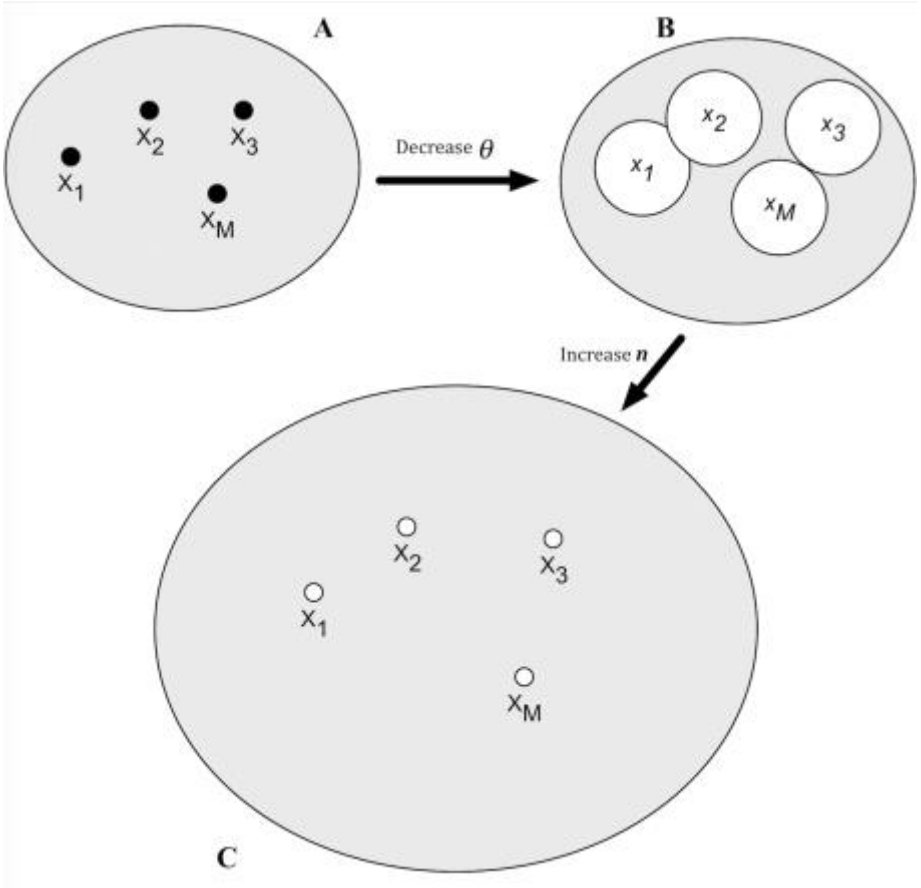

图 4 这张图说明了降低匹配阈值 θ 的概念差异。大的灰色区域表示所有可能的 SDR 空间，这里的元素 $x_1, x_2, \ldots x_M$ 都是区域内的 SDR 个体。在空间 A，我们看到精确匹配的场景，这里的 $\theta = w$ 并且 SDR 都是空间里单独的点。当你减小 θ 时，潜在匹配的集合增大了。注意，空间 B 中的 $x_1, x_2, \ldots x_M$ 现在是区域内更大的圆片，表示相比在空间 A，会有更多的 SDR 在空间 B 匹配上它们。因为你减小 θ 时，白色区域相对于灰色区域的比例会变得很大，有更大的可能性随机发生错误匹配。空间 A 和空间 B 的大小一样，因为它们共有固定的向量模 n。如果我们增大 n，也就是增大潜在 SDR 的空间。白色区域相对于灰色区域的比例会变小，如空间 C 所示。从 A 到 B 再到 C 的转变说明了在参数 θ 和 n 间的折衷：减小 θ 带给你更多的健壮性，也会增大你对错误匹配的灵敏程度，但是再增大 n 可以减轻这个不利影响。

采用合适的参数来衡量 SDR 可以拥有健壮性来抵抗大量噪声，也能够保持很小的误报概率。为了收敛到期望的参数值，我们需要计算误报的似然性作为匹配阈值的目标优化函数。

给定一个 SDR 编码 x 和另一个随机 SDR 编码 y，两者有相同的向量模 n 和向量势 w，发生错误匹配的概率是多少？也就是 $\text{overlap}(x, y) \geq \theta$ 的可能性是多少？匹配定义为有 θ 个或者更多个比特重叠，最多是 w 个。总共有 $\binom{n}{w}$ 个不同的编码方式，发生误报的概率是：

$$fp_w^n(\theta) = \frac{\sum_{b=\theta}^{w} |\Omega_x(n,w,b)|}{\binom{n}{w}} \tag{4}$$

如果 $\theta = w$，或者表示精确匹配的话会发生什么？等式 4 中的分子计算结果等于 1，并且该等式退化为等式 2。

为了对等式 4 有更直观的感觉，再次假设向量参数里的向量模 $n = 1024$ 且向量势 $w = 4$，如果阈值 $\theta = 2$，对应 50%的噪声，那么发生误差的概率是 14587 分之 1。就是说，当有 50%的噪声时，有很大的可能性会发生错误匹配。如果 w 和 θ 分别增大到 20 和 10，错误匹配的概率会急剧地减小到不足 10^{13} 分之 1！因此，保持 n 不变的同时，适当增大 w 和 θ，可以取得非常完美的健壮性，以便应对多达 50%的噪声。图 5 说明了 HTM 在实际中使用的值。

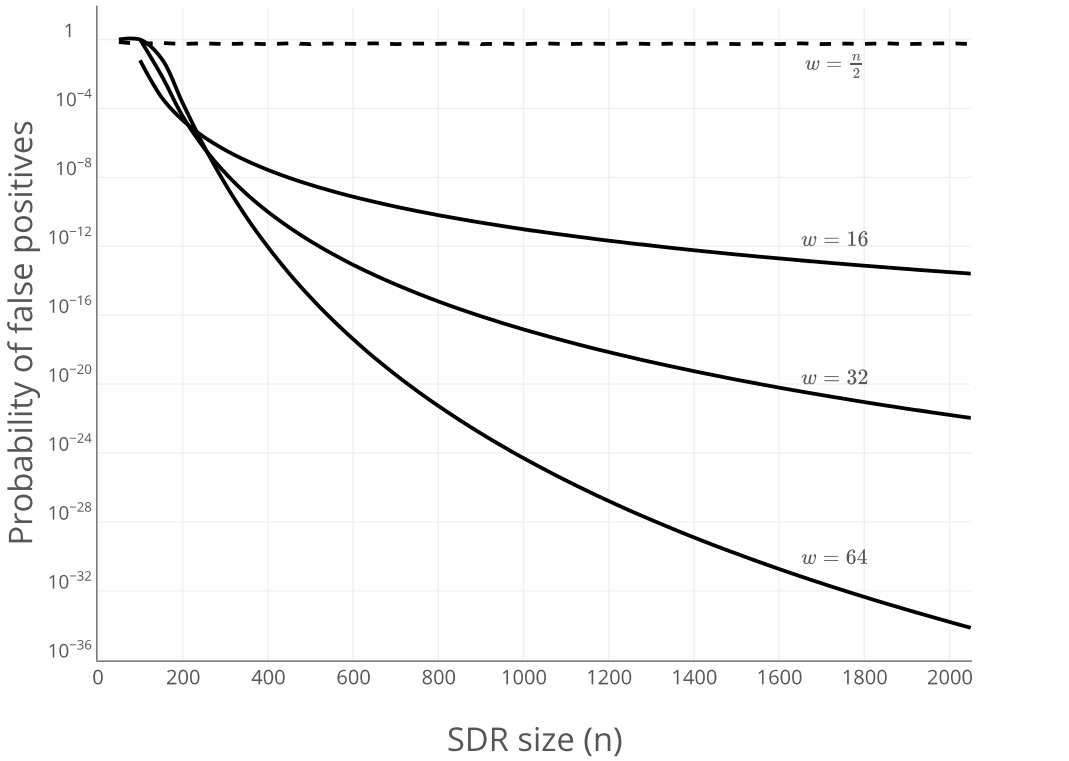

图 5 这张图说明了等式 4 参数间的作用关系。三条实曲线表示了当你增大 SDR 的模 n 时，误差率会快速下降（也就是误报率）。每条曲线代表不同数目的 ON 比特，即向量势 w 的情况，它们都有恒定为 50%的匹配阈值 θ。当 n 增大时，这三条曲线的误差下降速度比指数级下降还要快，当 $n > 2000$ 时误差基本为 0。虚线表示当 SDR 中的半数比特是 ON 时的误差率。注意，这条线保持着相对较高的误差率（大约为 50%），意味着如果采用非稀疏表征，不可能获得抵抗噪声的健壮性。要实现低误差率，稀疏性和高维度都是必要的。

子采样

SDR 的一个有趣特征是能够可靠地对比向量的子采样描述。也就是通过匹配大尺寸模式中激活比特的较小子集识别大尺寸的分布模式。假设 x 是一条 SDR，x' 是 x 的子采样描述，这样有 $w_{x'} \leq w_x$。不言而喻，只要满足 $\theta \leq w_{x'}$，子采样描述 x' 总会匹配 x。但是，如果你增大子采样，误报概率也会增大。

那么 x' 与随机的 SDR y 发生错误匹配的概率是多少？这里的重叠集是对子采样描述 x' 进行计算的，而非完整的向量 x。如果 $b \leq w_{x'}$ 并且 $b \leq w_y$，那么与 x' 正好有 b 个比特重叠的模式数目为：

$$\left|\Omega_{x'}(n, w_y, b)\right| = \binom{w_{x'}}{b} \times \binom{n - w_{x'}}{w_y - b} \tag{6}$$

给定阈值 $\theta \leq w_{x'}$，那么误报概率为：

$$fp_{w_y}^{n}(\theta) = \frac{\sum_{b=\theta}^{w_{x'}} |\Omega_{x'}(n, w_y, b)|}{\binom{n}{w_y}} \tag{7}$$

注意，在比较的向量中，等式 6 不同于等式 3，等式 7 也不同于等式 4。换句话说，子采样仅仅只是之前讨论的非精确匹配特征的变体。

举个例子，假设 $n = 1024$ 并且 $w_y = 8$ 。对 x 中的半数比特做子采样，并且把阈值设为 2（也就是 $w_{x'} = 4$ 并且 $\theta = 2$ ），我们会发现错误概率是 3142 分之 1。但是，把 w_y 增大到 20 并且按比例调整相关参数（也就是 $w_{x'} = 10$ 并且 $\theta = 5$ ），误报概率会急剧下降到 250 万分之 1。继续增大直到 $n = 1024$ 、 $w_{x'} = 20$ 、 $w_{x'} = 20$ 、 $\theta = 10$ ，这也是 HTM 参数更贴近实际使用的取值，误报概率会骤降至 10^{12} 分之 1！考虑到阈值大约是最初的 ON 比特数目的 25%，这个效果非常显著。图 6 说明了子采样对应到不同 HTM 参数的可靠性。

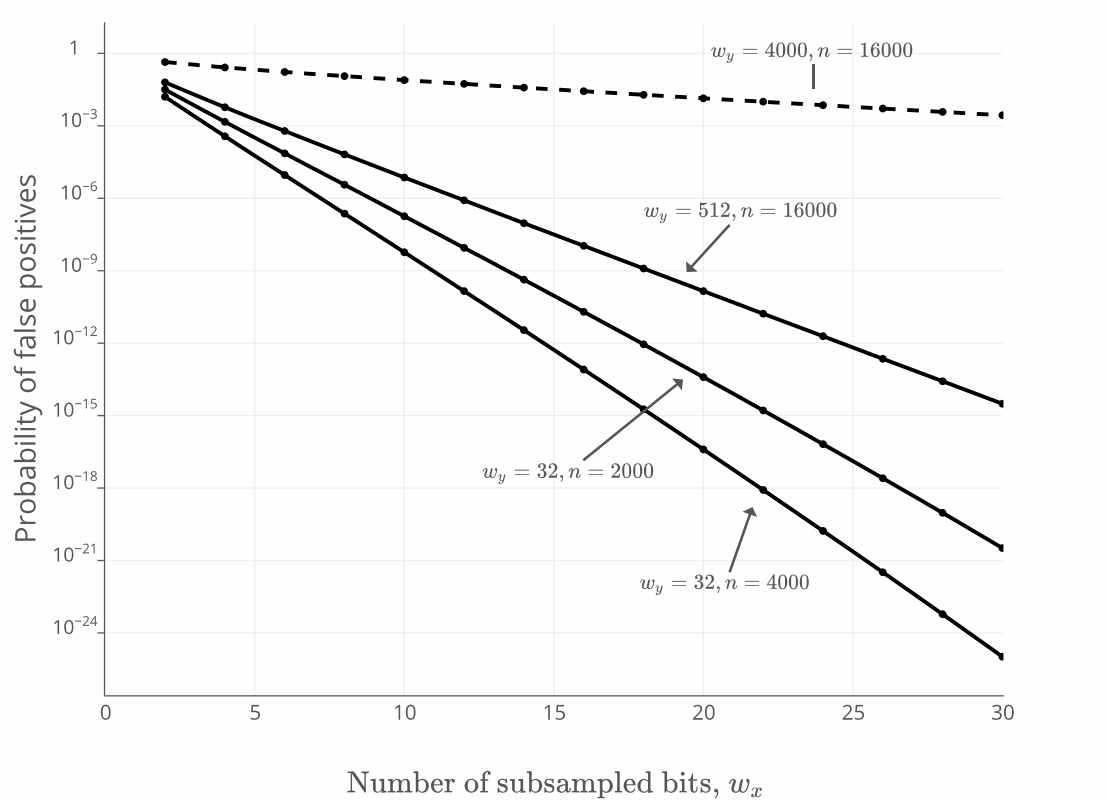

图 6 这张图说明的是等式 6 的情况，这里我们把误差率（也就是误报率）看作是关于子采样大小（也就是比特数目）的函数。三条实曲线代表几个特定的维度和稀疏度，表示了随着子采样比特数目的增大，误差率产生指数级的改善。如果有充分高的维度和稀疏度，采用 15 到 25 的子采样值可以达到非常低的误差率。相反，虚线代表的是一个相对稠密的表征（25%的 ON 比特）所对应的误差率；尽管维度很高，但是误差也很高。

采取实用参数，明显 SDR 会产生最小的误报概率。子采样和非精确匹配的特征允许我们把 SDR 用作可靠的分类机制，这会在下一节论述。

对一列 SDR 向量的可靠分类

SDR 的一个有益的用途是对向量进行分类，我们可以可靠地分辨某条 SDR 是否属于一组相似的 SDR。我们考虑一种类似最近邻域分类的分类形式。假设 X 是包含 M 个向量的集合，$X = \{x_1, \ldots, x_M\}$，其中每个向量 x_i 都是一条 SDR。给定随机的 SDR y，我们判断其是否属于该集合的依据如下：

$$\exists_{x_i \in X} match(\boldsymbol{x}, \mathbf{y}) = true \tag{8}$$

我们对被噪声污染了的向量进行分类的可靠性如何呢？更具体来说，如果我们从向量 x_i 的 n 个比特中随机选出 t 个并且切换它们的 ON / OFF 来引入噪声，那么发生误报分类的似然是多少？假设 $t \leq w - \theta$ ，并且这里没有漏报，只有误报。那么这个问题就变成了“对随机向量 y 的分类是误报的概率是多少？”。因为相对于匹配来讲， X 中的所有向量都是独特的，所以发生误报的概率是：

$$fp_x(\theta) = 1 - (fp_w^n(\theta))^M \tag{9}$$

因为发生单个重叠的误报概率非常小，所以使用以下的约束条件更加实用：

$$fp_x(\theta) \leq Mfp_w^n(\theta) \tag{10}$$

考虑如果所有向量的模 $n = 64$ 并且势 $w = 3$。如果 $\theta = 2$，你的列表里存储了 10 个向量，误报率大约是 22 分之 1。增大 w 到 12 并且增大 θ 到 8，保持比值 $\frac{\theta}{w} = \frac{2}{3}$ 不变，误报率会下降到大约 2363 分之 1。现在把参数增大到更实际的取值：$n = 1024$ 和 $w = 21$，并且 $\theta = 14$（也就是 w 值的 2/3）。在这种情况下，这 10 个向量的误报率会骤降至 10^{20} 分之 1。实际上，采用这些参数值，存储十亿个向量的误报率要比 10^{12} 分之 1 更低！

这个结果说明了 SDR 的一个显著特征。假设一个大的模式集合都是用 SDR 编码的，并且存储在列表里。那么其中很大数目的模式都可以接近完美地检索出来，甚至有大量噪声的情况下。设定 SDR 的参数 n、w 和 t 的重要原则是它们需要充分的大。正如上面的例子说明的，如果取较小的值比如 $n = 64$ 和 $w = 3$，你的 SDR 会不能利用好这些特征。

SDR 的合并

SDR 最迷人的特征之一是能够通过对所有的向量做 OR 操作从而把一组模式可靠地存储到一个固定的表征。我们称之为合并特征。为了存储由 M 个向量组成的集合，合并机理仅仅只是对全部的向量做布尔 OR 操作来产生一个新向量 X。要判断是否新的 SDR y 属于这个集合，我们只需要计算 $\text{match}(X, y)$。

$$\boldsymbol{x}_1 = [01000000000010000000 \ldots 010]$$
$$\boldsymbol{x}_2 = [00000000000000000010 \ldots 100]$$
$$\boldsymbol{x}_3 = [10100000000000000000 \ldots 010]$$
$$\vdots$$
$$\boldsymbol{x}_{10} = [00000000000000110000 \ldots 010]$$

$$\boldsymbol{X} = \boldsymbol{x}_1 OR \boldsymbol{x}_2 OR, \ldots, \boldsymbol{x}_{10}$$

$$\boldsymbol{X} = [11100000000110110000 \ldots 110]$$

$$\boldsymbol{y} = [10000000000001000000 \ldots 001]$$

$$\therefore match(\boldsymbol{X}, \boldsymbol{y}) = 1$$

图 7（顶部） 对包含 M 个 SDR 向量的集合做 OR 操作得到合并向量 $\mathbf{X}$。因为每个独立的向量都有 2%的 ON 比特，并且 $M = 10$，所以断定向量 $\mathbf{X}$ 中 ON 比特所占比重最多是 20%。这背后的逻辑并不复杂：如果这组向量彼此没有重叠，那么合并向量中的每个 ON 比特都会对应它们自身的 ON 比特，所以把 ON 比特的比重累加即可。如果存在重叠，那么合并向量中的 ON 比特都会与之对应，所以 ON 比特的比重会更低。 *（底部）* 计算 $\text{match}(\mathbf{X}, \mathbf{y})$ 的过程揭示了 $\mathbf{y}$ 是否属于合并向量 $\mathbf{X}$ 表示的集合，也就是说，$\mathbf{y}$ 中 ON 比特的位置对应到 $\mathbf{X}$ 中的位置也是 ON 比特。

合并特征的优势在于一个固定大小的 SDR 向量可以存储一个动态的集合。像这样，细胞的固定集合及其连接运作于一个动态列表。它也提供了分类操作的替补方案。在 HTM 中，SDR 的合并被广泛用于时序预测，以便进行时序池化、表现不变性以及创造高效的层次结构。但是，能够可靠地存储在一个集合中的向量的数目是有限的。换句话说，合并特征有着增大误报率的劣势。

合并特征有多可靠？没有漏报的风险；如果给定的向量在集合中，它的所有对应位置的比特都会是 ON，无论其他模式是什么样的，重叠是完美的。但是，合并特征会增大误报的似然。如果向量的数目 M 充分地大，那么集合的合并向量的 ON 比特会趋于饱和，判断任意的随机向量几乎都会得到误报匹配。理解这种关系是很很有必要的，这样我们可以规避合并特征的局限。

让我们先来计算精确匹配时，也就是 $\theta = w$ 时的误报概率。这种情况下，如果一个随机模式 y 的所有比特都与合并向量 X 重叠，就发生了误报。当 $M = 1$ 时，合并向量中任意给定的比特是 OFF 的概率是 $1 - s$，这里 $s = \frac{w}{n}$。当 M 增大时，误报概率是：

$$p_0 = (1-s)^M \tag{11}$$

经过 M 次合并操作，合并向量 X 中任意给定的比特是 ON 的概率是 $1-\mathrm{p}_0$ 。所以误报概率，也就是随机模式 y 中所有 w 个比特是 ON 的概率是：

$$p_{fp} = (1-p_0)^w = (1-(1-s)^M)^w \tag{12}$$

用来得到等式 12 的技术类似于分析布隆过滤器时对误报率的推导【Bloom, 1970; Broder and Mitzenmacher, 2004】。细微的差别在于，布隆过滤器中的每个比特都是独立选取的，也就是重置抽样。这样的话，给定的向量可能包含不足 w 个 ON 比特。在我们的分析中，保证了在每个向量中都恰好有 w 个 ON 比特。

上面的推导过程让我们搞清楚了，在某些条件下，我们可以通过合并来存储 SDR 并且不用担心发生误报。比如，考虑设置 SDR 的参数为模 $\mathrm{n}=1024$ 和势 $\mathrm{w}=2$ 。存储 20 个向量也就是 $\mathrm{M}=20$ ，发生误报的概率大约是 680 分之 1。但是，如果 w 增大到 20，概率会急剧地下降到大约 55 亿分之 1。这是合并特征的一个显著特征。实际上，如果 M 增大到 40，误差发生的概率仍然会优于 10^{-5} 分之 1。

为了对合并特征有一个直观的感觉，合并向量中 ON 比特的期望数目是 $\mathrm{n}(1-\mathrm{p}_0)$ 。如等式 12 所示，它增长得会比线性更慢；额外的合并操作会在作为结果的 SDR 上产生越来越少的 ON 比特。考虑 $\mathrm{M}=80$ 的情况，这里 20%的比特都是置 0 的。如果我们考虑一个额外的包含 40 个 ON 比特的向量，那么它有适当的概率会对应这 20%中的至少 1 个比特，所以它不会是误报。也就是说，只有那些 w 个 ON 比特都对应在合并向量中 80%的 ON 比特位置上的向量才是误报。随着我们增大 n 和 w，可以共同进行 OR 操作的模式的数目也会可靠并且充分地增加。如图 8 所示，如果 n 和 w 充分地大，即使 M 增大，误报的概率也会保持在能够接受的较低水平上。

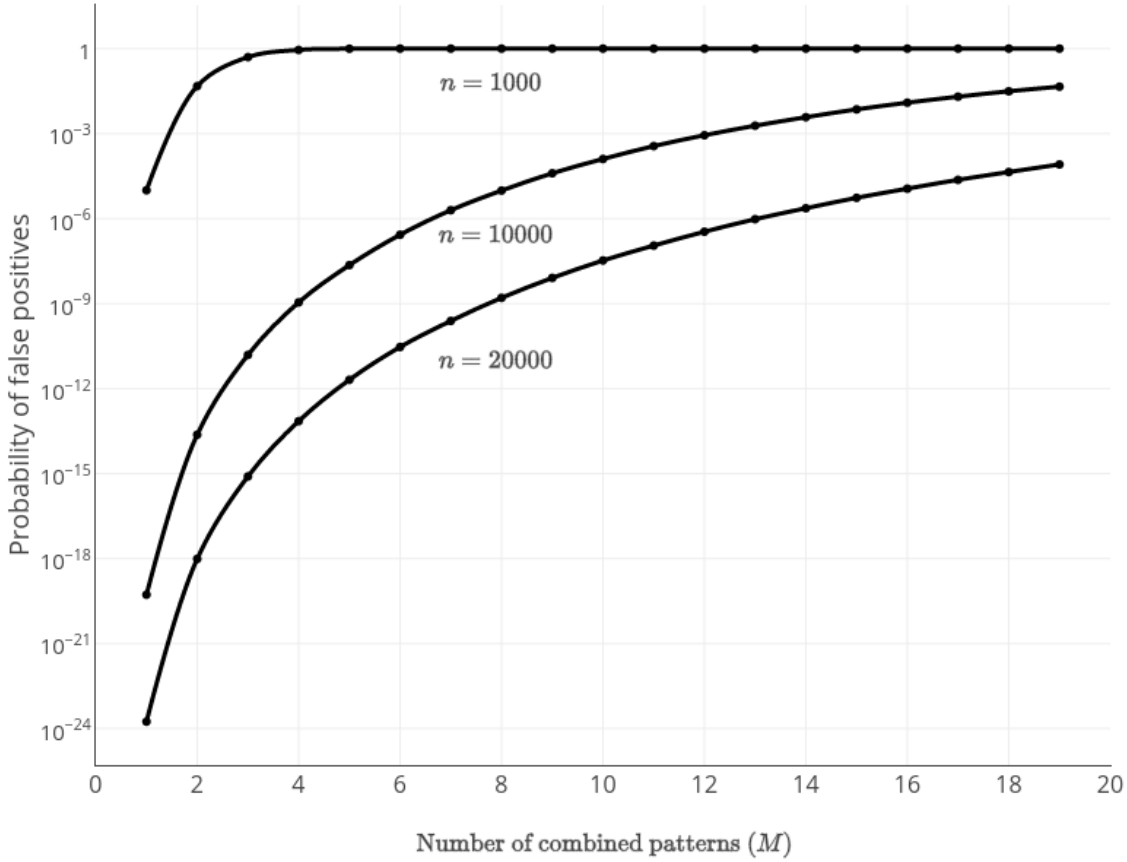

图8 本图表现了等式12的分类误差率（也就是匹配SDR的合并集合的误报率）。这三条线表现了SDR的几个维度在 $\mathrm{w}=200$ 时的

计算值。我们看到随着存储模式的数目的变化，误差率单调增大。更重要的是，本图说明了SDR的大小是关键因素：少量的比特（1000个比特）导致相对较高的误差率，而更大的向量（10000以上的比特）更加健壮。可以在网上交互地操作这张图：plot.ly网址的占位。

SDR 的合并在噪声环境下的健壮性

正如上面提到的，合并向量 X 中 ON 比特的准确数目是 $\tilde{w}_X = n(1-p_0)$，这里我们使用了波浪线表示法来代表合并向量。假设 $n \geq \tilde{w}_X \geq w$，我们可以计算出重叠集合的期望大小：

$$E[|\Omega_X(n,w,b)|] = \binom{\widetilde{w_X}}{b} \times \binom{n-\widetilde{w_X}}{w-b} \tag{13}$$

如果发生匹配，我们需要大小为 θ 个或者更多比特（最多是 w 个）的重叠。所以发生错误匹配的概率是：

$$\varepsilon \approx \frac{\sum_{b=\theta}^{w} |\Omega_X(n,w,b)|}{\binom{n}{w}} \tag{14}$$

注意等式 14 是误差率的近似值，因为我们正在处理合并向量 X 中 ON 比特的期望数目[1]。

正如你所预料的，误差率会随着阈值降低而增大，但是这种折衷的影响可以通过增大 n 来减轻。假设向量模 $n = 1024$ 并且向量势 $w = 20$。如果存储 20 个向量也就是 $M = 20$，精确匹配时的误报概率大约是 50 亿分之 1。当阈值 $\theta = 19$ 时误报率会增大到 1.23 亿分之 1。当阈值 $\theta = 18$ 时误报率会增大到 400 万分之 1。但是当阈值 $\theta = 18$ 保持不变，如果你增大 n 到 2048，误报率会急剧地下降到 2230 亿分之 1！这个例子说明了 SDR 的合并特征对噪声的健壮性，而且它还是我们另一个更大的主题的例子：SDR 长度的细微线性变化可以对误差率产生超指数的改善。

运算效率

尽管 SDR 向量很大，但是所有我们论述过的运算耗时都在关于 ON 比特数目的线性范围内[2]。也就是说，运算复杂度依赖于 ON 比特的数目 w，而不是向量的模 n。这对 HTM 系统很重要，因为实际上 $w \ll n$。另外，因为向量是二进制的，所以可以在大多数 CPU 上快速运算。但是实际情况没有这么理想，因为采用了更标准的距离度量，复杂度通常是 $O(n)$，此外还要包括浮点运算。

在接下来的一节中，我们会论述人脑是如何利用 SDR 的数学特征的，以及它是如何在 HTM 系统中体现的。

脑和 HTM 系统中的 SDR

稀疏分布表征和它们的数学原理应该如何与人脑中的信息存储和检索联系起来？本节我们会罗列 SDR 运用于脑中的一些方式，以及对应在 HTM 理论中的部分。

神经元的激活形式是 SDR

如果你观察新皮质中的部分神经元，你会发现它们的活动会是稀疏的，其中有很低比重的神经元是处于高度激活（传递脉冲）的状态，剩余的神经元是抑制的或者很缓慢地传递脉冲。SDR 代表了一组神经元的活动，置 1 和置 0 比特分别表示激活和相对抑制的神经元。HTM 系统的所有功能都是基于神经元和 SDR 之间的这个基本关联。

生物神经元的活动比简单的 1 或者 0 更加复杂。神经元传递脉冲，脉冲在某种意义上是二进制的输出信息，但是不同类型的神经元和在不同情况下它的频率和模式会有很大差别。关于如何解释神经元的输出信息有不同的观点。一个激进的观点认为，每个独立脉冲的节奏是很重要的，峰电位的时机对信息进行了编码。其他的理论家认为神经元的输出是一个标量值，与脉冲速率对应。但是，已经证明了新皮质可以如此迅速地执行重要任务，以至于参与的神经元甚至都没有足够的时间处理每个神经元的第二个脉冲进而促成任务的完成。在那些任务中，峰电位的节奏和脉冲速率不能解释编码信息。有时神经元会在进入稳定的脉冲速率前开始一个传递两到四个快速交替的脉冲的“微爆”。这些“微爆”会对突触后的细胞，也就是接收输入信息的细胞产生长期而且持久的影响。

HTM 认为神经元会处于数个状态之一：

---

[1] 这个近似背后的假设是，ON 比特的实际数目是和期望数目相等的。

[2] 运算复杂度是 $O(n)$，独立于向量的长度 n。

- 激活（传递脉冲）
- 抑制（没有传递脉冲或者很缓慢地传递脉冲）
- 预测（去极化后和传递脉冲前）
- 预测后激活（传递脉冲之前的“微爆”）

这些 HTM 神经元状态不同于其他的那些神经网络模型，我们会对这个给出一些解释。首先，众所周知，一些生物神经元会根据它们接收的输入信息和理想的“感受野”的匹配情况按照不同的速率传递脉冲。但是，单个的神经元对神经网络的表现不是非常重要的；激活神经元的数目才是最重要的，任何单个的神经元都可以停止运作，对网络的影响不大。由此可见，可变的脉冲速率对新皮质的功能而言不是必要的，所以在 HTM 模型中我们可以忽视这个特征。我们总是可以通过在一条 SDR 中使用更多比特来弥补可变编码的缺失。迄今为止所有构建的 HTM 实现都在没有可变速率编码的情况下运作得很好。回避可变速率编码的第二个理由是二进制的细胞状态让软件以及硬件实现容易得多。HTM 系统几乎不需要浮点运算，而这在任何包含速率编码的系统中都是需要的。没有浮点数学的需求，用来实现 HTM 的硬件会简单得多。有一个对可编程计算机的类比：当人们最初开始制造可编程计算机时，一些设计师提倡十进制的逻辑。二进制的逻辑胜出了，因为它更容易制造。

尽管 HTM 神经元没有可变速率的输出信息，它们确实纳入了两个不存在与其他理论中的新状态。当神经元识别出顶树突或者底树突的模式时，树突产生一个局部的 NMDA 脉冲，可以对细胞体去极化而无需产生体脉冲。在 HTM 理论中，细胞内部的去极化状态代表了对未来活动的预测，在序列记忆中起到关键作用。最后，在一些情况下，神经元会通过“微爆”脉冲启动。一种可以引起“微爆”的情况是当细胞从一个之前的去极化状态启动的时候。“微爆”会活化突触后神经元的代谢性受体，这会引起长时间的去极化和学习效果。尽管神经科学还没有对“微爆”有定论，但是 HTM 理论需要系统在输入信息不同于预测时表现得不一样。“微爆”和代谢性受体可以扮演这个角色。通过施加代谢作用，神经元可以在它的输入信息终止后保持激活，并且能够表现出增强的学习效果。这两个状态，预测（去极化）和预测后激活（“微爆”）是关于 HTM 理论是如何结合自上而下的系统层级的理论需求和详细的生物学细节进而获得对生物学的新见解的优秀案例。

图 9 脑中 SDR 的可视化（占位）

神经网络预测是 SDR 的合并

HTM 序列记忆会对将要发生的情况作出多个同步的预测。这种能力是 SDR 合并特征的一个实例。对应到生物学是多组神经元细胞（SDR）同时去极化。因为每个表征都是稀疏的，所以很多预测可以同步进行。比如，考虑实现 HTM 序列记忆的一层神经元。如果其中的 1%是激活状态并且产生 20 个不同的预测，那么会有大约 20%的神经元进入去极化／预测状态。即使有 20%的神经元去极化，系统仍然能够可靠地排查是否有预测发生了。作为人而言，你不会意识到那些预测，因为预测状态的细胞没有传递脉冲。但是，如果发生了意外的输入信息，网络可以侦查它，你就会察觉到有什么不对劲。

突触是一种存储 SDR 的方式

计算机内存一般称作“随机存取存储器”。字节存储在硬盘驱动器或存储器芯片上的某个存储器位置。要存取字节的值，你需要知道它在存储器中的地址。“随机”这个词的意思是，只要你知道需要检索的信息的地址，你就可以按照任意顺序检索它。脑中的存储器称作“联想存储器”。在联想存储器中，一条 SDR 与另一条 SDR 相连，相连的这一条还会与其他的相连，联系会一直持续下去。SDR 通过与其他 SDR 的联系会回忆起来。没有中心化的存储器或者随机存取。每个神经元都会参与形成 SDR 和学习关联。

考虑一个神经元，如果我们想要识别一个特定的活动模式。为了达到这个目的，神经元会与模式中的激活细胞形成突触。如上所述，一个神经元只需要形成少量的突触，通常会小于二十个，以便在大量细胞中准确地识别出模式，只要模式是稀疏的。形成新突触是脑中的几乎所有记忆的基础。

但是我们不是只想要一个神经元来识别一个模式；我们想要一组神经元来识别一个模式。这样，一条 SDR 会唤起另一条 SDR。我们想要 SDR 模式 A 唤起 SDR 模式 B。只要模式 B 中的每个激活细胞组成二十个突触连接上模式 A 中任意的一些细胞，这就可以实现。

如果 A 和 B 两种模式是在同一个神经元群体中前后相继的模式，那么习得的从模式 A 到模式 B 的关联是一个过渡，形成了序列记忆的基础。如果模式 A 和模式 B 在不同的神经元群体中，那么模式 A 会同时激活模式 B。如果模式 A 中的神经元连接到模式 B 中的远端突触，那么模式 B 就是预测的模式。如果模式 A 中的神经元连接到模式 B 中的近端突触，那么模式 B 就是激活神经元组成的激活模式。

所有的联想存储器的操作都使用相同的基本记忆机制，也就是在神经元的树突段形成新突触。因为所有的神经元都有数十个树突段和数以千计的突触，每个神经元不仅仅识别一种模式，而是数十种独立的模式。每个神经元都会参与形成很多不同的 SDR。

小结

SDR 是脑的语言，同时 HTM 理论规定了如何创建、存储、回忆 SDR 和 SDR 序列。本章我们学习了 SDR 强大的数学特征，以及这些特征是如何让脑和 HTM 能够学习和归纳序列的。

参考文献

Kanerva, P. (1988). Sparse Distributed Memory. Bradford Books of MIT Press.

Kanerva, P. (1997). Fully distributed representation. Proceedings of 1997 Real World Computing Symposium (RWC '97, Tokyo, Jan. 1997), pp. 358–365. Tsukuba-city, Japan: Real World Computing Partnership.

Bloom, B.H. (1970). Space/ Time Trade-offs in Hash Coding with Allowable Errors. Communications of the ACM, Volume 13, Number 7, July 1970, pp. 422-426.

Broder, A., & Mitzenmacher, M. (2004). Network Applications of Bloom Filters, A Survey. Internet Mathematics, Volume 1, No. 4, pp. 485-509.

Ahmad, S., & Hawkins, J. (2016). How do neurons operate on sparse distributed representations? A mathematical theory of sparsity, neurons and active dendrites. *arXiv*, 1601.00720. Neurons and Cognition; Artificial Intelligence. Retrieved from http://arxiv.org/abs/1601.00720