

胶囊网络原理

Joewell He

2018/8/22

1 前言

胶囊网络是Hinton在论文“Dynamic Routing Between Capsules”上提出的一种针对对象识别任务所设计的网络结构，该论文于2017年发表在人工智能顶级会议NIPS上。胶囊网络自问世后就引来了许多争议。Hinton曾大力批判用于传统神经网络的反向传播算法(Back propagation)的不合理性，进而提出用胶囊网络与其动态路由算法(Dynamic Routing, a.k.a routing-by-agreement mechanism)来替代传统神经网络与其反向传播算法。然而，有人也曾质疑Hinton这一说法，他们指出，在Hinton最新提出的胶囊网络上，仍然不可避免的要使用反向传播算法。不管学术上争议如何，胶囊网络的问世确实给深度学习带了一些新的思考。胶囊网络和动态路由算法的原理并不容易理解，加上论文原文的可读性确实差强人意，使得人们对动态路由的原理和做法有许多疑问。本文从多个角度阐述胶囊网络，总结了一些对胶囊网络的思考和改进。由于作者并非是计算机视觉领域的研究者，故本文没有从图像和视觉的角度解释胶囊网络，而是更多的关注于胶囊网络本身的性质。限于作者水平有限，难免有错漏之处，欢迎专家和读者给予批评指正，任何建议请发送邮件到hejiawei@hnu.edu.cn。

Contents

1	前言	1
2	胶囊网络(CapsNet)是什么?	3
2.1	胶囊(Capsule)	3
2.2	胶囊背后的直觉	4
2.3	动态路由算法	5
2.4	CapsNet结构及性能分析	9
3	胶囊的聚类理论解释	13
3.1	动态路由的聚类直观理解	13
3.2	模糊聚类	14
3.3	动态路由的聚类分析	15
3.4	改进的动态路由	18
4	胶囊网络的NLP应用与展望	20
4.1	胶囊文本分类模型	21
4.2	胶囊模型改进策略	22
4.3	胶囊网络NLP应用展望	23

2 胶囊网络(CapsNet)是什么？

2.1 胶囊(Capsule)

在介绍胶囊之前，我们先回顾一个神经网络最基本的单元——神经元。一个神经元就是一个带有激活函数的线性分类器。通常我们用一个 n 维的向量 $x \in R^n$ 来表示一个样本，其中样本的每一维 x_i 代表样本的某个特征。那么神经元就用下式表示：

$$f = \text{activate}(\langle w, x \rangle + b) \quad (1)$$

其中 w 代表神经元的权重， b 是偏置项， $\langle \cdot, \cdot \rangle$ 是内积运算， $\text{activate}(\cdot)$ 是激活函数，常见的激活函数有relu, sigmoid, softmax等。图1是一个神经元的示意图，可以看到，神经元用一个标量对样本特征进行描述，并且输出也是一个标量。这也就是所谓的scalar in scalar out。

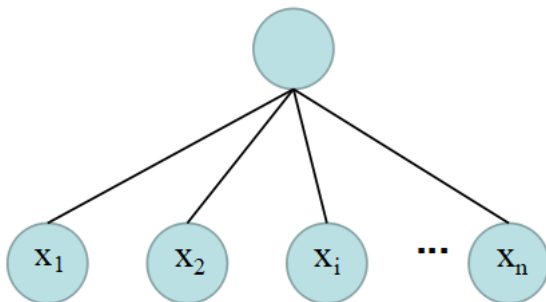


图 1: 神经元

其实早在2011年，Hinton就曾初步的提出了胶囊的概念[4]。胶囊作为一种替代神经元的新结构，扩展了对样本特征的描述。胶囊用一个向量来表示样本的一个特征，简单的说，一个胶囊就代表了一个特征向量。例如，样本 X 有 m 个特征，每个特征我们用一个 n 维的向量来表示，那这个样本就可以表示为 $x = (c_1, c_2, \dots, c_m)$ ，其中 $c_i \in R^n$ ，代表一个胶囊(特征向量)。很显然，用一个向量来表示一个特征，所能表达的特征信息量要远多于一个标量。与神经元类似，上层胶囊与下层胶囊也是互联的，下层胶囊向上层胶囊输送特征，上层胶囊对收到的一组特征信息进行筛选组合，抓取关键的一些信息。图2展示了一个上层胶囊 c_j 与下层胶囊 (c_1, \dots, c_m) 之间的互联，其中每个胶囊是一个3维的特征向量。可以看到，与神经元scalar in scalar out不同，胶囊是一种vector in vector out的机制。

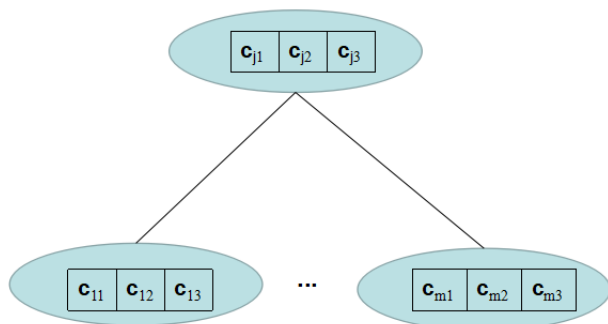


图 2: 胶囊

可是，vector in vector out是不是一种新的机制呢？当然不是！如果我们了解RNN(包括LSTM, GRU等)就知道，RNN同样可以读取一组向量，输出一组向量。并且，在NLP领域中，用RNN做vector in vector out的操作是十分常见的，例如seq2seq，基于LSTM的语言模型等都是RNN做vector in vector out。这样说来，我们不禁又会有疑问：胶囊与RNN有何区别？胶囊的工作机制到底是什么呢？

2.2 胶囊背后的直觉

胶囊网络是受到人类视觉识别物体的原理的启发而创造的。Hinton认为，人类识别一个物体的时候，会忽略一些不相关的细节，将视觉中枢神经在观看物体时获取到的物体关键部分的信息送给上层的中枢神经做更高层的决策。与之相似，当我们将图像的特征表示为胶囊，上层的胶囊**并不会完整的接受每一个下层胶囊的全部信息，而是有差别的接受一部分突出的信息**[6]。Hinton将这种上层胶囊接受下层胶囊的输入进行决策的过程描述为一种树形(parse tree[3])的过程。

图3展示了一个2层胶囊网络的例子，这个网络包含2个上层胶囊和3个下层胶囊。我们假设上层胶囊只关注一个人的头发特征和衣服特征，并且根据这两个特征对人进行分类。在上层的胶囊中，左边的胶囊关注长头发和穿裙子的人，认为拥有这种特征人是女人；右边的胶囊关注短头发和穿牛仔裤的人，认为这样的人是男人。红色的线表示某一次决策中，一个穿着裙子，留着长头发，有樱桃小嘴的人的图像特征从下层胶囊到上层胶囊的转化过程。每一个下层胶囊的信息会分成多份，传到上层胶囊中，并且，会把自身更多的传递到与自身一致性更高的上层胶囊中(下几个章节我

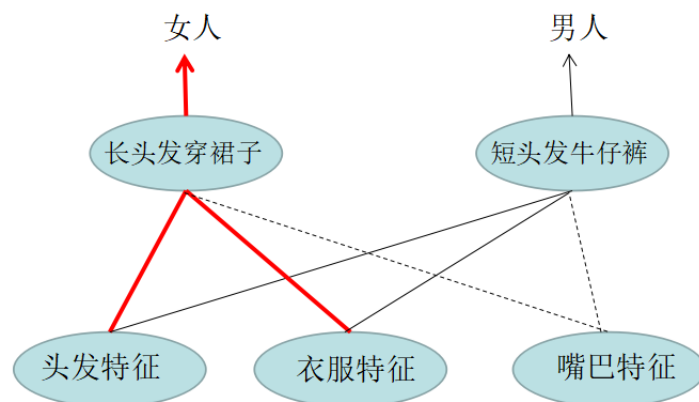


图 3: 树形的决策过程举例

们会详细介绍并推导如何获得这样的一致性)。在这个例子中，头发的特征是长头发，那么他会把自身的一大部分传递给上层中左边的胶囊，一小部分传递给上层右边的胶囊。假设这个特征含有的信息量是1，他可能传给上层左边的胶囊0.8，上层右边的胶囊0.2(我们将红线画成更粗的线，也是想表达了这层含义)。而对于嘴巴特征，上层中并没有与它一致性很高的胶囊，所以很遗憾，它的信息几乎不会传递给上层(我们用虚线来表示它传递了很少的信息)。

在图2中，红色的线代表有大量信息传递的线。一个上层的胶囊，如果获得了下层胶囊大量的信息，我们称它为一个**被激活的胶囊**，也把这个信息传递的过程称为**激活过程**。就像图2一样，这种激活过程，一般会形成一种树形的决策过程，也就是说特定的特征只会激活胶囊网络每一层的一小部分胶囊，就像Hinton在论文中所说的，“特定的特征所形成的一颗树是从一个固定的多层胶囊网络中雕刻出来的，就像雕塑是从岩石上雕刻出来的一样”。

那么，如何让下层胶囊激活与自身一致性更大的胶囊呢？这需要用到一种叫做**动态路由**的算法。接下来，我们将更正式的定义胶囊网络，并阐述动态路由的原理。

2.3 动态路由算法

一个胶囊作为一个特征向量，有这样的性质：它的方向代表了某个特定的特征，它的模长(L2范数)代表了这个特征存在的可能性大小，也就

是特征存在的概率，并且所有胶囊的模长总是小于1的。我们用一个两层的胶囊网络为例，来介绍动态路由是如何让信息从下层胶囊到上层胶囊传递的。

设有胶囊网络的下层有 m 个胶囊，记为 u ，第上层有 n 个胶囊，记为 v 。其中 v_j 代表第 j 个上层胶囊， u_i 代表第 i 个下层胶囊。每一个胶囊都含有一个自身的权值矩阵，用来抽取下层胶囊中自身所关注的信息，这些矩阵的值是可以通过训练来得到的(实质上，我们可以将它理解为一个全连接的神经网络)。对于从 u_i 与 v_j 的连接，我们先用 v_j 的权值矩阵 W_{ij} 来做一个线性映射，使

$$u_{j|i} = W_{ij}u_i \quad (2)$$

$u_{j|i}$ 我们可以认为它是下层胶囊 u_i 关于上层胶囊 v_j 的一个副本，每个下层胶囊并不会直接把自身传给上层，而是利用这个副本进行传递。直观上，我们认为经过线映射的下层胶囊副本将更突出这一上层胶囊所关注的特征信息(一个经过训练的神经网络，通常可以做到这一点)。每个下层胶囊 u_i 与上层胶囊 v_j 之间会有一个匹配权值(coupling coefficients)，我们记为 c_{ij} ，且满足 $\sum_{j=1}^n c_{ij} = 1$ ，用它来决定下层胶囊对上层胶囊信息传递的多少。图4描述了这种副本信息传递的过程，其中 $c_{11} + c_{12} + c_{13} = 1$ 。

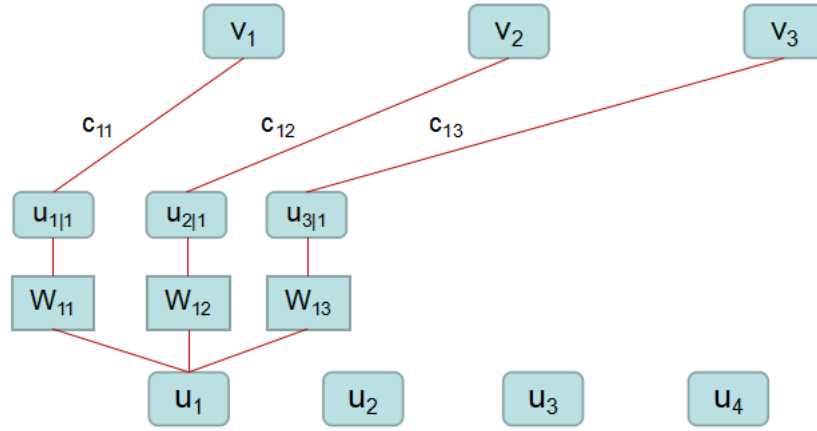


图 4: 下层胶囊对上层胶囊信息传递过程

每个上层胶囊 v_j ，需要接收来每个自下层胶囊副本的信息，这个过程

我们用如下的式子描述:

$$\begin{aligned} s_j &= \sum_{i=1}^m c_{ij} u_{j|i} \\ v_j &= \text{squash}(s_j) \end{aligned} \quad (3)$$

s_j 是胶囊 v_j 接受来自下层胶囊副本信息的加权和, 值得注意的是, 这些权值的和并不为1。回顾之前对于胶囊性质的描述, 胶囊的模长代表特征存在的大小, 所以我们引入 $\text{squash}(\cdot)$ 函数对 s_j 缩放, 以获取最终的上层胶囊 v_j 。 $\text{squash}(\cdot)$ 定义如下:

$$\text{squash}(s_j) = \frac{\|s_j\|^2}{1 + \|s_j\|^2} \frac{s_j}{\|s_j\|} \quad (4)$$

这个函数可以使得特征和 s_j 的方向不变, 即胶囊所描述的特征信息不变, 同时让胶囊 v_j 的模长不超过1。我们还注意到, $\text{squash}(\cdot)$ 函数中用到了一个缩放函数 $\frac{x}{1+x}$, 这个函数在 x 取值大时, 接近1, 即不进行缩放; 当 x 的值接近0时, 它也接近0, 即对长度小的 s_j 进一步缩小。我们用图5描述上层胶囊接收副本信息的过程, 我们再次强调 $c_{11} + c_{21} + c_{31} + c_{41} \neq 1$

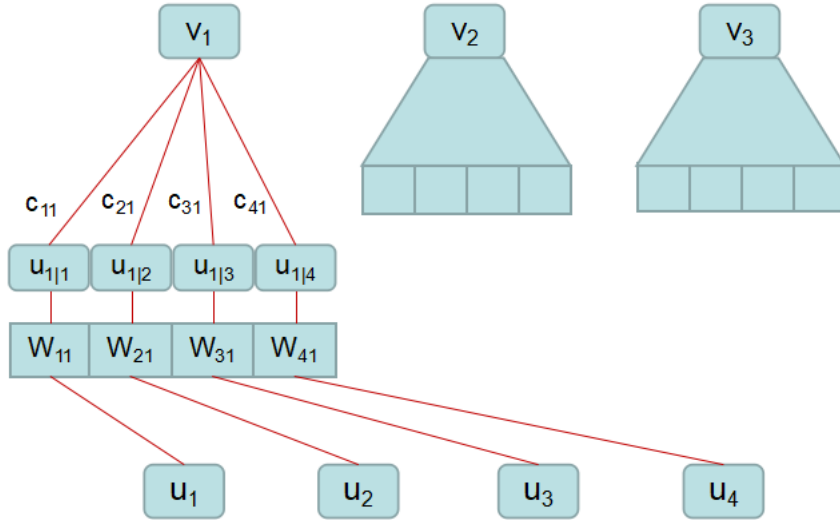


图 5: 上层胶囊对下层胶囊信息的接收过程

到这里, 我们已经了解了上下层胶囊是如何传递特征信息。然而仍然存在一个问题, 我们如何确定权值矩阵 C ? 我们已经知道 C 是一个 $m \times n$ 的

矩阵, $c_{ij} \in C$ 表示下层胶囊 u_i 向上层胶囊 v_j 传递信息的匹配权值, 且对于任意下层胶囊 u_i 满足 $\sum_{j=1}^n c_{ij} = 1$ 。对于 c_{ij} , 我们希望, 当 $u_{j|i}$ 与 v_j 越相近, 匹配程度越高时, 它的值越大; 反之, 则越小。而 v_j 又是通过对 $u_{j|i}$ 加权, 然后缩放而得到的, 加权后的权重来自于矩阵 C 。于是乎我们陷入了一个鸡生蛋, 蛋生鸡的问题: 要求 C , 需要知道上层胶囊 v , 要生成上层胶囊 v , 需要一个权重矩阵 C 。解决这个问题的关键, 就是Hinton所提出的动态路由算法。

Algorithm 1 动态路由算法

Input:

$u_{j|i}$ /*底层胶囊副本*/
 m /*底层胶囊数量*/
 n /*上层胶囊数量*/
 r /*路由迭代次数*/

Output:

v_j /*上层胶囊 v_j */

```

1: for  $i$  from 1 to  $m$  do
2:   for  $j$  from 1 to  $n$  do
3:      $b_{ij} = 0$ 
4: for  $r$  iterations do
5:   for  $i$  from 1 to  $m$  do
6:      $c_i = \text{softmax}(b_i)$ 
7:   for  $j$  from 1 to  $n$  do
8:      $s_j = \sum_{i=1}^m c_{ij} u_{j|i}$ 
9:      $v_j = \text{squash}(s_j)$ 
10:  for  $i$  from 1 to  $m$  do
11:    for  $j$  from 1 to  $n$  do
12:       $b_{ij} = b_{ij} + \langle u_{j|i}, v_j \rangle$ 
return  $v_j$ 

```

在算法1中, 引入了一个中间量 b_{ij} 。起始时, 对任意的 i, j , $b_{ij} = 0$, 且对任意 i , $c_i = \text{softmax}(b_i)$ 。于是易得到:

$$\begin{aligned} c_{i1} &= c_{i2} = \cdots = c_{in} \\ c_{1j} &= c_{2j} = \cdots = c_{mj} \end{aligned} \tag{5}$$

也就是说下层胶囊 u_i 将信息等概率的传送给每一个上层胶囊。上层胶囊 v_j

也等权重的接受每一个下层胶囊的信息。我们直观上对动态路由的过程做一个简单的理解，对于上层胶囊 v_j ，无妨设起始时权重：

$$c_{1j} = c_{2j} = \cdots = c_{mj} = w \quad (6)$$

那么易得第一次路由迭代时：

$$\begin{aligned} s_j &= mw(u_{j|1} + u_{j|2} + \cdots + u_{j|m}) \\ v_j &= \text{squash}(s_j) \end{aligned} \quad (7)$$

又知 $\text{squash}(\cdot)$ 不改变特征向量方向，则 v_j 与 $u_{j|i}$ 的均值 $(u_{j|1} + u_{j|2} + \cdots + u_{j|m})/m$ 是共线的。无妨设 $v_j = k\bar{u}_{j|i}$ ，其中 $\bar{u}_{j|i}$ 是 $u_{j|i}$ 的均值， k 是常数项，则对于 $\langle u_{j|i}, v_j \rangle$ 有：

$$\begin{aligned} \langle u_{j|i}, v_j \rangle &= \langle u_{j|i}, k\bar{u}_{j|i} \rangle \\ &= \frac{k}{m} \langle u_{j|i}, (u_{j|1} + u_{j|2} + \cdots + u_{j|m}) \rangle \\ &= \frac{k}{m} (\|u_{j|i}\|^2 + \sum_{i \neq k} \langle u_{j|i}, u_{j|k} \rangle) \end{aligned}$$

于是 $\langle u_{j|i}, v_j \rangle$ 的值，在一定程度上，与 $\|u_{j|i}\|$ 的大小是相关联的。回顾我们对胶囊的定义，胶囊的模长代表了特征存在与否，也就是说当第一次更新 b_{ij} 时， $u_{j|i}$ 模长越长， b_{ij} 增长的越快，于是对应的匹配权值 c_{ij} 就会逐步变大。这就使得 v_j 的方向会偏向那些模长较长的特征。接下来的迭代过程中， v_j 与模长较长的特征方向相近，使得内积的值不断增大，而那些与 v_j 方向接近正交，甚至相反的特征，内积会接近0，或负值，使得 b_{ij} 增长缓慢，或者缩小。此消彼长下，这样的迭代过程，就会让上层胶囊，更关注下层胶囊中某些存在可能性更高的特征。Hinton经过一些实验验证，认为将迭代次数 r 设为3到5已经可以取得较好的收敛性。

当然，这个理解实际上是不严谨的。上式中给出的推导不能证明内积项 $\langle u_{j|i}, v_j \rangle$ 完全依赖于 $\|u_{j|i}\|$ ，只能用它作为一种直观的理解方式。此外，我们无法推至下层胶囊 u_i 经过线性映射后的副本 $u_{j|i}$ 的模长，是否仍然严格与其特征存在性存在对应关系(这种对应关系表现为：模越长，存在性越高，反之，存在性低)。我们只能经验上认为经过训练的权重矩阵 W_{ij} 用来对 u_i 进行线性映射，会让这一特性更加突出。在之后的章节，将会对动态路由做更严谨的理论分析。

2.4 CapsNet结构及性能分析

Hinton设计了一个简单的3层CapsNet，如图6，在基准数据集MINIST上测试了其性能，本小节简单的介绍其网络结构，效果分析，以及损失函数

的设计。

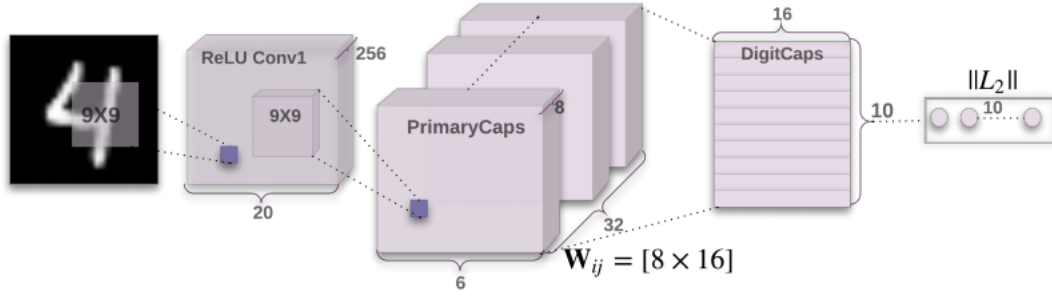


图 6: 用于MINIST的三层胶囊网络结构

卷积层：该层直接与输入图像(28×28)相连，为网络的最低层。该层有256个9×9的卷积核，于是图像经过卷积层后，会变成20×20×256的特征图。

原始胶囊层¹(PrimaryCaps)：该层有32个原始胶囊，每个原始胶囊中有8个步长为2的，9×9×256的卷积核。于是对于输入的20×20×256的特征图，在每个原始胶囊中被转换成6×6×8的特征图，这一层一共有32个这样的特征图。我们已经知道，胶囊代表一个特征向量，而怎么用表示特征图呢？Hinton将特征图的每个坐标点，当成一个特征，于是每个原始胶囊有36个8维的特征，原始胶囊层总共有 $32 \times 36 = 1152$ 个8维的特征向量。需要注意的是，动态路由过程只在两个胶囊层中发生，卷积层和原始胶囊层没有动态路由过程。

数字胶囊层(DigitCaps)：该层是最上层的胶囊，共有10个胶囊，每个胶囊 v_j 代表CapsNet将手写图片中的数字识别为数字 j ，其中 $0 \leq j \leq 9$ ， v_j 的模长代表图片中的数字是数字 j 的概率。这一层胶囊为16维，于是对于每个胶囊 v_j ，都有一个1152个16×8的权值矩阵，将8维的特征转换为16维的副本，然后再做动态路由过程(也可以说是8×16维的矩阵，取决于你将向量左乘还是右乘)。

损失函数：对于 n 分类问题，最上层胶囊会有 n 个胶囊，对于每一个类别 k ，定义其损失如下：

$$L_k = T_k \max(0, m^+ - \|v_k\|)^2 + \lambda (1 - T_k) \max(0, \|v_k\| - m^-)^2 \quad (8)$$

$T_k = 1$ 表示训练样本属于第 k 类， m^+ 与 m^- 在Hinton的实验中分别设置为0.9与0.1，代表一种分类概率分布的置信度。当样本属于第 k 类时，损

¹许多资料译为主胶囊层，作者认为此处译为原始胶囊层更合适

失函数要求 v_k 的模长越接近0.9越好，当样本不属于第 k 类，损失函数要求 v_k 接近0.1。这与最大似然的思想很相近，要求样本属于正确类别的概率高的同时也要求样本属于错误类别的概率低。其中 λ 的作用是调整错误类的损失在总损失中的占比，如果 λ 过大，错误类的损失主导了损失函数，就会让所有胶囊的特征模长都趋向于变得更小，所以调整 λ 为合适的值在实际训练中和十分重要。对于一个样本，总损失为：

$$L = \sum_{k=1}^n L_k \quad (9)$$

解码层：Hinton让分类正确的样本的胶囊通过3个全连接层的网络，将胶囊解码成图片，这个结构如图7所示。例如，样本属于第 k 类，且胶囊对其分类正确，即 v_k 的模长是最大的，就让 v_k 经过一个3层的全连接层，还原成 28×28 维的图片。这个过程十分简单，不作过多赘述。

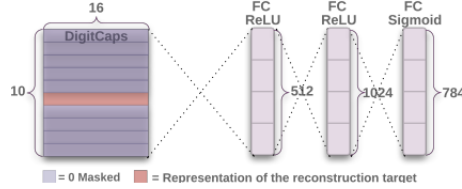


图 7: Decode层：将胶囊解码成图片

效果分析：Hinton在这个简单的3层胶囊网络上做了许多的实验，这里不一一介绍。其中CapsNet对于MINIST的分类误差低至0.25%，当然，MINIST作为基准数据集，简单的全连接网络也能达到很好的分类效果，而胶囊内置了许多全连接层，单纯以这个结果判断胶囊网络和动态路由的有效性说服了似乎有些欠缺。不过论文提供了解码层的对于胶囊信息的解码的实验，达到了一定的效果。如图8所示， l 表示样本的label， p 表示胶囊的预测值， r 表示通过 v_p 重构的图片。最右的两张图混淆了3和5，这无可厚非，我们可以看到即便是肉眼也较难区分图片中的3和5。当然，Hinton还做了其它诸如重叠对象分割等有趣的实验，有兴趣的读者可以自行阅读论文原文。

共享权重的胶囊：许多人认为一个上层胶囊对每个下层胶囊都有一个权重矩阵 W ，会使得训练时参数量巨大，于是有人提出每个上层胶囊只内嵌一个权重矩阵，用来对所有下层胶囊进行线性映射。我们将这种上层胶囊对每个下层胶囊共享同一个权重矩阵的结构，称为共享权重的胶囊。图9













(l, p, r)	(2, 2, 2)	(5, 5, 5)	(8, 8, 8)	(9, 9, 9)	(5, 3, 5)	(5, 3, 3)
Input						
Output						

图 8: 胶囊信息解码效果

是普通全连接的胶囊网络和共享权重的胶囊网络的对比，可以看到，共享权重的胶囊明显减少了需要训练的参数量。且经过一些实验验证，共享权重的胶囊在大多数时候表现效果与全连接胶囊的差距不大，相较由于减少参数而增加的训练效率来说，更多人倾向于选择共享权重的胶囊。

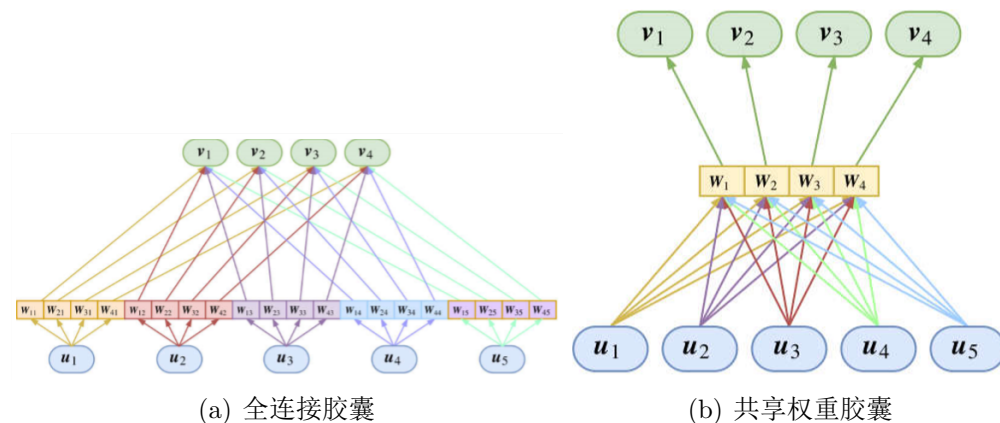


Figure 9: 全连接VS共享权重

总的来说，胶囊网络提供了一种向量表示特征的方式，同时给出了用于上下层网络激活的动态路由算法。但是，胶囊网络尚处于起步阶段，在理论上还有许多不严谨的地方，例如，迭代次数 r 的选择，在论文中只给出了实验性结果；求解 C 和 v_j 的迭代方式，Hinton本人并未给出理论依据。有幸的是，在最新的人工智能顶级学术会议ICLR上，已经有人聚类理论补全了动态路由的理论支撑，在下一章节我们将详细介绍动态路由的聚类理论依据。

3 胶囊的聚类理论解释

胶囊网络的动态路由过程，Hinton只给出了直观上的解释，并没有给出一个严谨的理论分析。所以在之前的阐述中，我们也仅做了一些直观上的理解。国内中山大学苏剑林发表了3篇技术博客^{2 3 4}，较早提出对动态路由的聚类解释，并给出了动态路由改进及基于k-means推导[2, 5, 1]。2018年3月，ICLR上新发表的文章“AN OPTIMIZATION VIEW ON DYNAMIC ROUTING BETWEEN CAPSULES”，对动态路由提出了更正式的优化目标定义，以及迭代算法解释，并在理论上改进了动态路由算法，使得算法本身在理论上更具有可解释性[7]。值得一提的是，虽然苏剑林的博客没有整理成论文发表在国际上，但是二者的思想具有很强的相似性，且苏剑林提出聚类解释时，该论文还没有公开。故作者认为聚类解释的理论完善是国内外学者共同提出的。

3.1 动态路由的聚类直观理解

直观上，胶囊的动态路由过程，与聚类过程有着很强的相似性。我们假设底层有 m 个胶囊，上层有 n 个胶囊，上层胶囊会偏向与自身相近的底层胶囊，这相当于将底层胶囊当成 m 个数据点，划分 n 个簇，进行聚类，其中每个上层胶囊就是聚类的簇中心。每一个上层胶囊都表征着底层一些有着共性的相似底层胶囊的中心，用这个簇中心代表新的特征以供更上层利用。图10是一个聚类过程，可以直观地认为簇中心就是3个上层胶囊的特征。

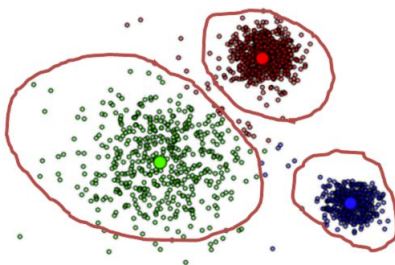


图 10: 聚类过程

²<https://kexue.fm/archives/4819>

³<https://kexue.fm/archives/5112>

⁴<https://kexue.fm/archives/5155>

3.2 模糊聚类

动态路由与普通的聚类的差别在于，动态路由用匹配权值 c_{ij} 来表示底层胶囊 u_i 属于上层胶囊 v_j 代表的簇的概率。而普通聚类则规定， u_i 属于第 j 簇时， c_{ij} 为1，否则为0，且一个底层胶囊 u_i 只能属于一个簇。这与动态路由的过程就有了明显的差别。

在对动态路由进行理论分析之前，我们先需要了解模糊聚类的概念。一般的聚类会在数据中给出若干个簇，将每个样本按簇聚在一起，认为这些样本属于这个簇。簇我们可以理解为类别，只是这个类别并不是数据一开始就有的，而是一种按照某种需求而划分的隐含的标签。所以在聚类中，我们很难判断聚类的好坏，因为从不同的角度和需求去聚类，会得到不同的结果。譬如对于一群学生，老师可能在某种情形下希望将男同学和女同学分开，这是一种聚类方式，也可能在另一种情形下，将高的同学和矮的同学分开，这又是另一种聚类的方式。一般的聚类认为数据属于某个簇是一种非0即1的确定逻辑。数据属于簇A就不可能属于其他簇B,C,D,...。然而在模糊聚类中，对一个数据，并不给出数据属于某簇的明确答案，只给出该数据属于这些簇的概率分布，譬如数据属于簇A的概率为0.8，属于簇B的概率为0.15，属于簇C的概率为0.03...。譬如我们将想将人群聚成2个类，簇的标准是秃与不秃，图11给出的样本就不能用百分百的确定去划分他们是秃还是不秃。

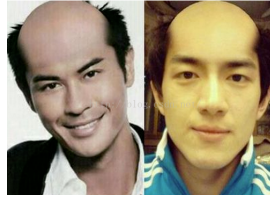


图 11: 秃与不秃的聚类

假设有 n 个数据的数据集 X ， $X = \{x_1, x_2, \dots, x_n\}$ ，其中数据都是存在于 p 维向量空间的点，即 $x_i \in R^p$ 。我们想用模糊聚类将数据划分为 c 个簇，那么就可以用一个矩阵 $U = R^{c \times n}$ 表示划分的结果。其中 U_i 是矩阵的第 i 列，它是一个 c 维的向量，代表了数据 i 属于个各簇的概率分布。且有约束：

$$\sum_{j=1}^c U_{ji} = 1 \quad (10)$$

3.3 动态路由的聚类分析

Dilin Wang与Qiang Liu在论文中指出，Hinton的动态路由其本质是优化一个带有KL散度正则项的模糊聚类损失函数。在介绍该损失函数前，先对引入KL散度的定义。

KL散度，又称相对熵，是用来衡量两个概率分布差异程度的统计量，设有原始概率分布 $p(x)$ 与观察概率分布 $q(x)$ ， $p(x)$ 对 $q(x)$ 的KL散度定义为：

$$D_{KL}(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)} \quad (11)$$

$$= \sum_x p(x) (\log p(x) - \log q(x)) \quad (12)$$

$$= E_x[\log p(x) - \log q(x)] \quad (13)$$

需要注意的是KL散度不具备对称性，即不满足 $D_{KL}(p||q) = D_{KL}(q||p)$

设有 n 个上层胶囊(即簇中心)，定义为 $S = \{s_1, s_2, \dots, s_n\}$ ，定义匹配权值矩阵 $C \in R^{m \times n}$ ，其中 c_{ij} 表示下层胶囊 u_i 属于第 j 个簇的概率(相当于之前定义的模糊聚类的划分矩阵的转置)，且满足：

$$\sum_{j=1}^n c_{ij} = 1 \quad (14)$$

Hinton的动态路由过程可以表示为优化以内积为相似度的带KL正则项的模糊聚类过程：

$$\begin{aligned} \min_{C, S} \{ \text{loss}(C, S) = - \sum_{i=1}^m \sum_{j=1}^n w_j c_{ij} \langle u_{j|i}, s_j \rangle + \sum_{i=1}^m D_{KL}(c_i || c_i^{old}) \} \\ \text{s.t. } c_{ij} > 0, \sum_{j=1}^n c_{ij} = 1, \|s_j\| \leq 1 \end{aligned} \quad (15)$$

其中 c_i 表示第 i 个底层胶囊关于 n 个簇的概率分布， c_i^{old} 是求解时上次迭代过程中的分布。优化的目标是使得 $\sum_{i=1}^m \sum_{j=1}^n w_j c_{ij} \langle u_{j|i}, s_j \rangle$ 足够大的同时， c_i 与 c_i^{old} 的分布的差异足够小。前者使得一个簇中心 s_j 确定后，分布 c_i 能更使离 $u_{j|i}$ 近的簇中心有更大的概率，这种“近”使用内积衡量而非距离，所以希望内积累加和最大化(如果用距离替换内积，那就应该最小化这一项)。最大化内积累加和与最小化距离累加和本质上是相似的。而KL散度正则化项使得迭代过程中 c_i 的分布趋向稳定。 w_j 是一个缩放系数，我们先暂且认为它是一个常量。还要注意的，在求内积时，我们使用 u_i 关于 s_j 线性映射的副本 $u_{j|i}$ 。

这是一个有不等约束的最小化问题，且同时需要优化两个目标。我们用坐标下降法来优化它，即依次固定 C 优化 S ，固定 S 优化 C 。对于有约束的最小化问题，我们写出它的拉格朗日函数：

$$L(C, S) = - \sum_i \sum_j w_j c_{ij} \langle u_{j|i}, s_j \rangle + \sum_i \sum_j c_{ij} (\log c_{ij} - \log c_{ij}^{old}) \\ + \sum_i \lambda_i (\sum_j c_{ij} - 1) + \sum_j \beta_j (\|s_j\| - 1) - \sum_{i,j} \gamma_{ij} c_{ij} \quad (16)$$

设它的最优解为 S^* ， C^* ，对于不等式约束优化的极值点，需要满足KT条件，即对任意的 i, j ：

$$\frac{\partial L}{\partial c_{ij}^*} = 0 \quad (17)$$

$$\frac{\partial L}{\partial s_j^*} = 0 \quad (18)$$

$$\beta_j^* (\|s_j^*\| - 1) = 0, \quad \beta_j^* \geq 0 \quad (19)$$

$$-\gamma_{ij}^* c_{ij}^* = 0, \quad \gamma_{ij}^* \geq 0 \quad (20)$$

$$c_{ij}^* > 0 \quad (21)$$

$$\sum_{j=1}^n c_{ij}^* = 1 \quad (22)$$

$$\|s_j^*\| \leq 1 \quad (23)$$

无妨先固定 C ，求解 s_j ：

$$\frac{\partial L}{\partial s_j} = - \sum_i w_j c_{ij} u_{j|i} + \frac{s_j \beta_j}{2\|s_j\|} = 0 \quad (24)$$

解得：

$$s_j = \frac{2\|s_j\| \sum_i w_j c_{ij} u_{j|i}}{\beta_j} \quad (25)$$

将其代入19式的约束，易知 β_j 为分母，不能为0，故约束项相当于求解：

$$\left\| \frac{2\|s_j\| \sum_i w_j c_{ij} u_{j|i}}{\beta_j} \right\| - 1 = 0 \quad (26)$$

解得

$$\beta_j = 2\|s_j\| \left\| \sum_i w_j c_{ij} u_{j|i} \right\| \quad (27)$$

将式27代入式25得

$$s_j = \frac{\sum_i c_{ij} u_{j|i}}{\left\| \sum_i c_{ij} u_{j|i} \right\|} \quad (28)$$

令

$$\hat{s}_j = \sum_i c_{ij} u_{j|i}$$

于是

$$s_j = \frac{\hat{s}_j}{\|\hat{s}_j\|}$$

这里便得到动态路由算法中 s_j 的更新公式。然后固定 S ，求解 c_{ij} ：

$$\frac{\partial L}{\partial c_{ij}} = -w_j \langle u_{j|i}, s_j \rangle + \log c_{ij} - \log c_{ij}^{old} + \lambda_i + 1 - \gamma_{ij} = 0 \quad (29)$$

易得：

$$\log c_{ij} = \log c_{ij}^{old} + w_j \langle u_{j|i}, s_j \rangle - \lambda_i - 1 + \gamma_{ij} \quad (30)$$

于是求得：

$$c_{ij} = \frac{c_{ij}^{old} e^{w_j \langle u_{j|i}, s_j \rangle + \gamma_{ij}}}{e^{\lambda_i + 1}} \quad (31)$$

又知存在约束 $\sum_{j=1}^n c_{ij} = 1$ ，即：

$$\sum_j \frac{c_{ij}^{old} e^{w_j \langle u_{j|i}, s_j \rangle + \gamma_{ij}}}{e^{\lambda_i + 1}} = 1 \quad (32)$$

则求得：

$$e^{\lambda_i + 1} = \sum_j c_{ij}^{old} e^{w_j \langle u_{j|i}, s_j \rangle + \gamma_{ij}} \quad (33)$$

带入到等式31，则求得：

$$c_{ij} = \frac{c_{ij}^{old} e^{w_j \langle u_{j|i}, s_j \rangle + \gamma_{ij}}}{\sum_k c_{ik}^{old} e^{w_k \langle u_{k|i}, s_k \rangle + \gamma_{ik}}} \quad (34)$$

对于拉格朗日常数项 r_{ij} ，还存在20式的约束条件，将34式代入到约束条件，易求得对于任意 i, j ， $r_{ij} = 0$ 。于是我们得到了 c_{ij} 的最优解：

$$c_{ij} = \frac{c_{ij}^{old} e^{w_j \langle u_{j|i}, s_j \rangle}}{\sum_k c_{ik}^{old} e^{w_k \langle u_{k|i}, s_k \rangle}} \quad (35)$$

令 $c_{ij} = e^{b_{ij}}$ ，带入35式，有

$$e^{b_{ij}} = \frac{e^{b_{ij}^{old}} e^{w_j \langle u_{j|i}, s_j \rangle}}{\sum_k e^{b_{ik}^{old}} e^{w_k \langle u_{k|i}, s_k \rangle}} \quad (36)$$

于是易导出 c_{ij} 的更新公式为

$$\begin{aligned} b_{ij} &= b_{ij}^{old} + w_j \langle u_{j|i}, s_j \rangle \\ c_{ij} &= \frac{e^{b_{ij}}}{\sum_k e^{b_{ik}}} \end{aligned} \quad (37)$$

只需要令缩放系数 w_j 为：

$$w_j = \frac{\|\hat{s}_j\|^2}{1 + \|\hat{s}_j\|^2}$$

则从坐标下降法求解带有KL散度正则项的模糊聚类问题的迭代过程，导出动态路由算法。至此，Hinton提出的动态路由算法的聚类解释已经推导完成。

然而，细心的读者会发现，在这个推导过程中有一个理论上并不严谨的地方。 w_j 依赖于 \hat{s}_j ，而 \hat{s}_j 与 s_j 相关，因此在固定 C ，求解 s_j 的过程中，将 w_j 当成常数项处理，是不合理的。而Hinton的动态路由算法中并没有给出 w_j 的更新规则，因此，原始的动态路由在理论上的确有欠缺之处。

3.4 改进的动态路由

原始的动态路由的确有理论上不完美的地方，然而科学研究的历史上，不乏有前人提出创新的做法，后人补充理论解释的案例。我们无法保证深度学习的算法都能向支持向量机一样，有着优雅的数学理论解释，但是Dilin Wang等人改进的动态路由算法使得胶囊网络在理论上的可解释性大大的提高。我们回顾关于动态路由的直观解释与聚类解释，有这么几个地方的分析在严格推敲下显得有些苍白无力。其一是聚类过程基于 $u_{j|i}$ ，而 $u_{j|i}$ 是经过权值矩阵映射的副本，我们无法保证经过映射后

的 $u_{j|i}$ 仍然保留有，方向是特征，模长代表概率的性质，也无法保证 $u_{j|i}$ 的模长是0到1之间的，因此当 $u_{j|i}$ 特别大时，会使簇中心偏向它，而忽略其他特征。其二是对于 w_j 项，优化过程中将其当成常数的做法，在理论上并不成立。

对于第一个问题，改进的动态路由在计算 u_i 的副本时，改为下式：

$$u_{j|i} = \frac{W_{ij}}{\|W_{ij}\|_F} u_i$$

其中 $\|W_{ij}\|_F$ 是权值矩阵 W_{ij} 的Frobenius范数。我们知道对于胶囊 u_i ，总是满足 $\|u_i\| \leq 1$ 。于是经过这样的线性映射后，副本 $u_{j|i}$ 模长的大小得到的限定。

对于第二个问题，Dilin Wang等人修改了优化目标，改为优化如下的模糊聚类损失：

$$\begin{aligned} \min_{C,S} \{ \text{loss}(C, S) = - \sum_{i,j} c_{ij} \langle u_{j|i}, s_j \rangle + \alpha \sum_{i,j} c_{ij} \log c_{ij} \} \\ \text{s.t. } c_{ij} > 0, \sum_{j=1}^n c_{ij} = 1, \|s_j\| \leq 1 \end{aligned} \quad (38)$$

最明显的当然是在优化目标中移除了 w_j 项。其次，它将KL散度正则化项改为用 c_i 的熵作为正则化项，这样做会使得 c_i 的分布更加均匀，平滑，从而使算法更加稳定。这也符合统计学上的最大熵原理，即在其它条件相同时，我们应该选择一种熵最大的分布作为随机变量的分布，以避免任何主观的原因影响了对于随机变量未知分布的估计。

求解它的方法同样是坐标下降法迭代的寻找不等式约束下的极值点，通过构建拉格朗日函数求KT点易得：

$$s_j = \frac{\sum_i c_{ij} u_{j|i}}{\|\sum_i c_{ij} u_{j|i}\|} \quad (39)$$

$$c_{ij} = \frac{e^{\frac{1}{\alpha} \langle u_{j|i}, s_j \rangle}}{\sum_i e^{\frac{1}{\alpha} \langle u_{j|i}, s_j \rangle}} \quad (40)$$

依上两式，可以导出改进的动态路由算法2。

算法2将缩放项放到的迭代之后，这让改进的动态路由在理论上更加优雅。并且对于所有 s_j ，使缩放系数的分母保持一致，从而使 w_j 更趋于稳定。此外，算法2引入了渐变系数 α ，用来调整熵正则化项在损失中的占比。直觉上来说，在迭代的早期，得到的聚类中心并不可靠，因此需要让 α 更大，以使 c_i 的分布更加趋于平滑，均匀。随着迭代的进行，聚类中心逐渐变得可靠，因此要调小 α 的值，让聚类损失主导损失函数。

Algorithm 2 改进的动态路由

Input: $u_{j|i}$ /*底层胶囊副本*/ m /*底层胶囊数量*/ n /*上层胶囊数量*/ r /*路由迭代次数*/**Output:** v_j /*上层胶囊 v_j */

```
for  $i$  from 1 to  $m$  do
  for  $j$  from 1 to  $n$  do
     $b_{ij} = 0$ 
  for  $r$  iterations do
    for  $i$  from 1 to  $m$  do
       $c_i = \text{softmax}(b_i)$ 
    for  $j$  from 1 to  $n$  do
       $\hat{s}_j = \sum_{i=1}^m c_{ij} u_{j|i}$ 
       $s_j = \frac{\hat{s}_j}{\|\hat{s}_j\|}$ 
    for  $i$  from 1 to  $m$  do
      for  $j$  from 1 to  $n$  do
         $b_{ij} = \frac{1}{\alpha} \langle u_{j|i}, s_j \rangle$ 
  for  $j$  from 1 to  $n$  do
     $w_j = \frac{\|\sum_i c_{ij} u_{j|i}\|}{1 + \max_k (\|\sum_i c_{ik} u_{k|i}\|)}$ 
  return  $w_j s_j$ 
```

4 胶囊网络的NLP应用与展望

虽然胶囊网络最初是为了应用于图像领域，胶囊的部分性质却与NLP的一些问题有着惊人的拟合程度。在NLP领域能轻易的找出vector in vector out 的任务，譬如文本分类，机器翻译，文本摘要等。自词向量问世以来，NLP领域多用一个连续的实值向量代表一个词，而NLP领域的处理对象却多为句子，文章。于是现阶段，用一组词向量来表示句子或文章，然后构建基于RNN，或CNN的模型，是很常见的做法。幸运的是，胶囊网络拥有处理一组向量输入的特性。这使得研究者和工程师们在面临NLP 问题时有了一种新的选择—胶囊网络。

Wei Zhao等人首次将胶囊网络应用于文本分类。他们对胶囊结构以及动态路由算法做了一些简单实用的改进，这里将介绍他们的改进。文本分类在NLP 中属于较为简单的任务，该团队在6个基准数据集上测试了胶囊网络的性能，将胶囊网络与几种强baseline，如LSTM、BiLSTM、CNN等作对比，并在4个基准数据集上获取了更好的性能。

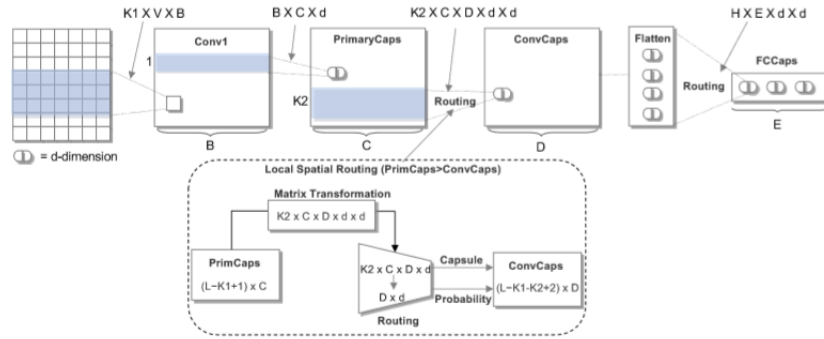


图 12: 胶囊文本分类模型构成

4.1 胶囊文本分类模型

Wei Zhao等人构建的胶囊文本分类模型由主要有4个层组成，它们分别是：n-gram卷积层，原始胶囊层，卷积胶囊层，全连接层。图12展示了这个模型，下面将对这个模型的各个层次分别介绍。为了与图对应，本文采用与原论文相同的数学符号。

n-gram卷积层：假设输入文本 $x \in R^{L \times V}$ ，其中 L 是文本的长度， V 是词向量的维度。n-gram卷积层将用 B 个卷积核 W^a 对输入文本进行卷积，用来提取n-gram特征。 $W^a \in R^{K_1 \times V}$ ， K_1 是n-gram窗口的长度。经过这一层得到的特征图 $M \in R^{(L-K_1+1) \times B}$

原始胶囊层：n-gram卷积层得到的特征图 M 的每一行 M_i 是一个 B 维的向量，包含了卷积层提取的n-gram信息。在原始胶囊层，我们进一步对信息做提取。用 $W^b \in R^{B \times d}$ 对每一个 M_i 进行转换，转换后的特征图 $p \in R^{(L-K_1+1) \times d}$ 。该层有 C 个这样的矩阵，于是得到 C 个这样的特征图 p ，将总共的特征图记为 $P \in R^{(L-K_1+1) \times C \times d}$ 。

卷积胶囊层：本层有两种做法，共享权重矩阵和非共享矩阵。由于本层的操作比较复杂，这里以共享权重为例，阐述本层的结构。两种做法的区别不大，非共享权重的细节可以由读者自行扩充。本层有 D 个胶囊，

且胶囊的维度与原始胶囊层相同，也是 d 维，于是对任意的第 j 个胶囊，权值矩阵 $W_j^{c1} \in R^{d \times d}$ ，记所有权值矩阵组成的张量 $W^{c1} \in R^{D \times d \times d}$ 。该层的胶囊并非普通的胶囊，可以理解为该层有 D 个大胶囊，每个大胶囊中有一些小胶囊，这些小胶囊共享权重矩阵 W_j^{c1} ，且只局部连接到 $K_2 \times C$ 的下层胶囊上，用这些胶囊做动态路由。原始胶囊层有 C 个经过提取的n-gram特征图 $p \in R^{(L-K_1+1) \times d}$ ，一个小胶囊与每个特征图 p 都连接起来，但是对于单个特征图 p ，它的局部连接只考虑了 K_2 个n-gram特征。这些小胶囊按照与卷积类似的方式，以 K_2 为窗口宽度，在 $L - K_1 + 1$ 的n-gram特征上滑动，步长为1。易计算知，每个大胶囊有 $L - K_1 - K_2 + 2$ 个小胶囊。每个小胶囊是一个 d 维的特征，共有 D 个大胶囊。于是这一层总共有 $(L - K_1 - K_2 + 2) \times D$ 个 d 维的小胶囊。需要注意的是，动态路由发生在每个小胶囊与该小胶囊连接的下层胶囊之间。

全连接层：本层也是胶囊层，设分类问题有 E 个类，本层就有 E 个胶囊，每个胶囊同样是 d 维的。该层的每个胶囊与卷积胶囊层的所有的 $(L - K_1 - K_2 + 2) \times D$ 个小胶囊连接起来，做动态路由。同样以共享权重为例，该层的权值矩阵张量为 $W^{d1} \in R^{E \times d \times d}$ 。

作者提供了2种参考的胶囊文本分类模型的结构，如图13所示。Capsule-A是一种基准结构，只需要设定n-gram的大小，然后将上述的4层网络叠加起来就可以得到Capsule-A模型。Capsule-B相当于Capsule-A的叠加，它借鉴了textcnn中设定不同n-gram窗口的方式，结合不同的n-gram大小，如bi-gram、tri-gram、four-gram，得到的Capsule-A结果，最后对结果做平均，得到Capsule-B模型。(图12，图13，为论文原文的配图，可读性有些欠缺，本文的后续版本会替换掉这两张图)

4.2 胶囊模型改进策略

这里同样提出了一些对动态路由的实用性改进策略。这些改进虽然不是通过理论推导得到的，却经过了一些实验验证，被发现能够提升模型性能。因此本文仅将它们当做调整模型的一些策略。

噪声类：文本分类中，通常有一些无关信息，例如停用词等，这些无关信息对文本分类的帮助并不大。可以在label中多增加一个噪声类，用以捕获这些无用信息，从而模型提高准确率。

部分抽样归一化：在动态路由中，下层胶囊 u_i 对上层胶囊的分布 $c_i = \text{softmax}(b_i)$ 决定了下层胶囊传递多少信息给每个上层胶囊。部分抽样归一化按一定的比例随机抽取部分上层胶囊，设这些被抽取的上层胶囊组成集

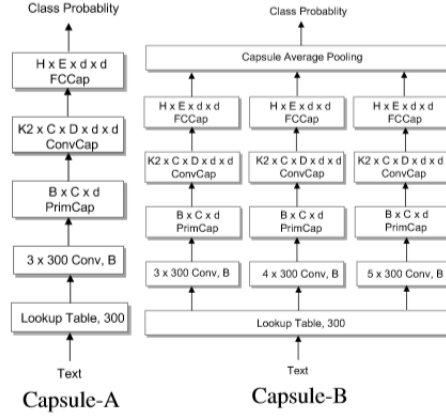


图 13: 分类模型结构

合 K ，则下层胶囊 u_i 对上层胶囊的分布 c_i 由下式确定

$$c_{ij} = \begin{cases} 0, & j \notin K \\ \frac{e^{b_{ij}}}{\sum_{k \in K} e^{b_{ik}}}, & j \in K \end{cases} \quad (41)$$

它将随机的屏蔽一些上层胶囊，与深度学习中的dropout有异曲同工之妙。

4.3 胶囊网络NLP应用展望

胶囊网络在层间传递向量这一机制，在NLP领域上有天然的优势。过去对于长文本，总以平均，或者加权和的方式将其表示为一个向量，用以做文本相似度，文本分类等任务。然而词，与长文本同样在向量的粒度上表示，对于长文本来说，一定会存在信息损失。胶囊网络则可以解决这个问题，如何用多个胶囊表示文本，用以实现更精确的文本相似度对比，语言模型等，值得人们思考探究。当然，长文本之间还存在序列关系，如何让胶囊网络学表示这种词之间的序列关系，从而设计出能够完成机器翻译，机器摘要任务的胶囊网络，也需要研究者做更多的钻研和努力。

参考文献

- [1] 三味Capsule: 矩阵Capsule与EM路由[<https://kexue.fm/archives/5155>]. 苏剑林. 2018.
- [2] 揭开迷雾, 来一顿美味的Capsule盛宴[<https://kexue.fm/archives/4819>]. 苏剑林. 2018.
- [3] Geoffrey E. Hinton, Zoubin Ghahramani, and Yee Whye Teh. Learning to parse images. In *International Conference on Neural Information Processing Systems*, pages 463–469, 1999.
- [4] Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51, 2011.
- [5] 再来一顿贺岁宴: 从K-Means到Capsule[<https://kexue.fm/archives/5112>]. 苏剑林. 2018.
- [6] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 3859–3869, 2017.
- [7] Dilin Wang and Qiang Liu. An optimization view on dynamic routing between capsules. In *6th International Conference on Learning Representations 2018*, 2018.