

Beijing Forest Studio  
北京理工大学信息系统及安全对抗实验中心



# 胶囊 (向量神经) 网络

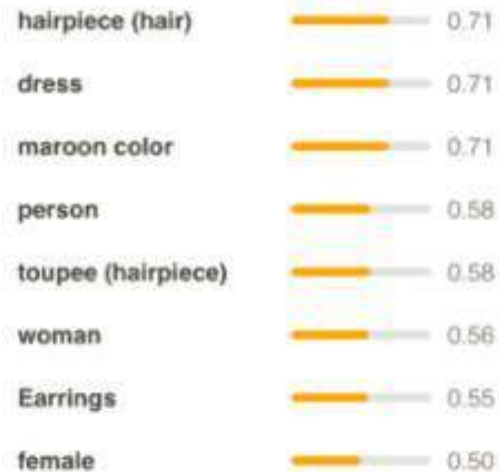
李筱雅 硕士

2018年02月23日

- 背景简介
- 基本知识
- 算法原理
- 网络结构
- 优劣分析
- 应用总结



# 背景简介



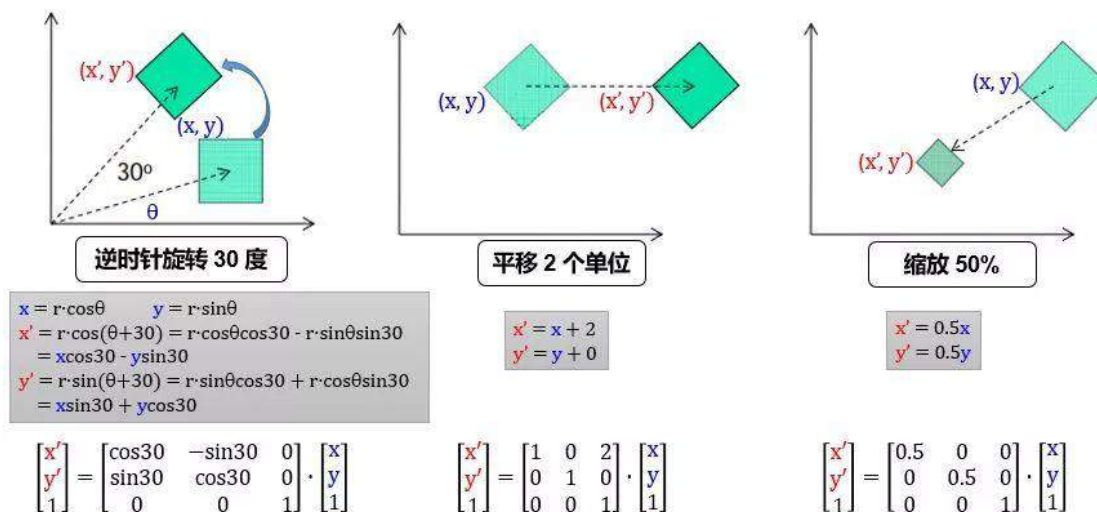
- CNN的缺陷：
  - 对物体之间的空间关系（spatial relationship）的识别能力不强
  - CNN对物体旋转之后的识别能力不强
- Hinton在2017年11月7日在《Dynamic Routing Between Capsules》中提出了Capsules。



# 基本知识

## • 姿态矩阵

– 姿态主要包括旋转、平移和放缩三种形式



• 将 (x,y) 先逆时针转30度，再向右平移2个单位，最后缩放50%到 (x',y') 可以由下列矩阵连乘得到

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_S \cdot \underbrace{\begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_T \cdot \underbrace{\begin{bmatrix} \cos 30 & -\sin 30 & 0 \\ \sin 30 & \cos 30 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_R \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

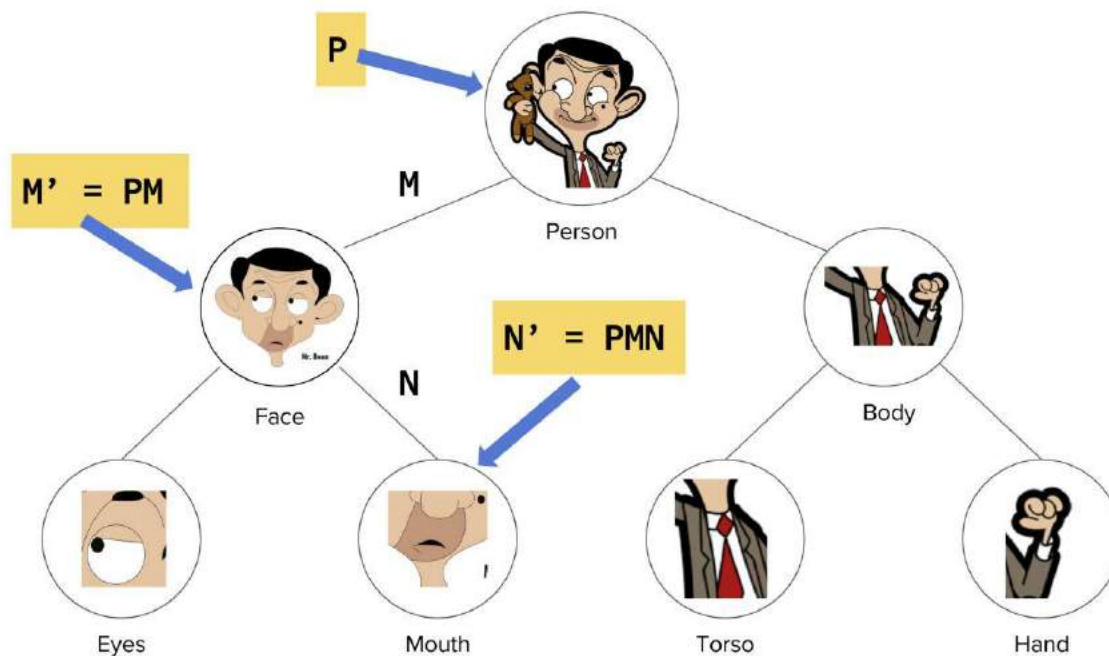
- 姿态矩阵
  - 延伸到3维空间

$$M = \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \leftarrow 2 \text{ 维姿态矩阵}$$

$$M = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \leftarrow 3 \text{ 维姿态矩阵}$$



- 姿态矩阵：该矩阵所定义的对象都是相对于照相机的视点



- 姿态矩阵P表示可以从相机看对象的不同视点

- 不变性和共变性

- 不变性 (invariance) : 表示不随变换变化

- 同变性 (equivariance) : 表示的变换等价于变换的表示



- 胶囊 (capsule) : 包含多个神经元的载体, 每个神经元表示了图像中出现的特定实体的各种属性。其中一个特殊的属性是图像中某类别的实例的存在, 输出数值的大小为实体存在的概率。
- 神经元类比:

		向量神经元 (VN)	标量神经元 (SN)
输入		$u_i$	$x_i$
操作	转换	$U_{j i} = W_{ij}u_i$	-
	加权总和	$s_j = \sum_i c_{ij}U_{j i}$	$a_j = \sum_i w_i x_i + b$
	非线性激活	$v_j = \frac{\ s_j\ }{1 + \ s_j\ ^2} \cdot \frac{s_j}{\ s_j\ }$	$h_j = g(a_j)$
输出		$v_j$	$h_j$



# 算法原理

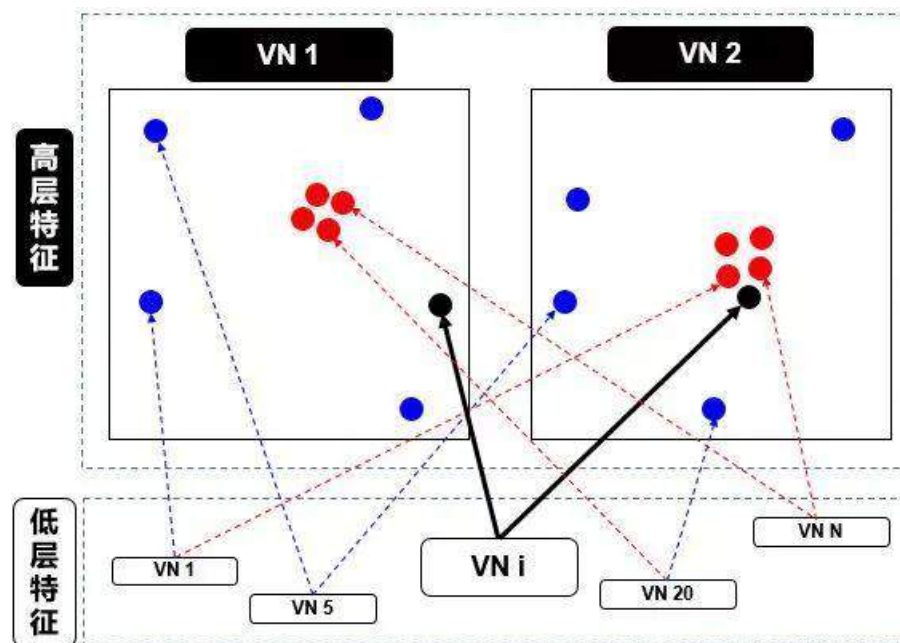
- 为使问题具体化，假设：
  - 上一层的VN代表眼睛（ $u_1$ ），鼻子（ $u_2$ ）和嘴巴（ $u_3$ ），称为底层特征
  - 下一层的第 $j$ 个VN代表脸（脸为其中一个），称为高层特征
- 第一步：矩阵转化

$$\underbrace{U_{j|i}}_{\text{由低层特征推出的高层特征}} = \underbrace{W_{ij}}_{\text{低层特征和高层特征的空间关系}} \cdot \underbrace{u_i}_{\text{低层特征}}$$
  
  
$$\begin{array}{l} \text{具体化} \rightarrow \left\{ \begin{array}{l} \underbrace{U_{j|1}}_{\text{由眼睛推出的脸的位置}} = \underbrace{W_{1j}}_{\text{眼睛和脸之间的空间关系}} \cdot \underbrace{u_1}_{\text{眼睛的位置}} \\ \underbrace{U_{j|2}}_{\text{由鼻子推出的脸的位置}} = \underbrace{W_{2j}}_{\text{鼻子和脸之间的空间关系}} \cdot \underbrace{u_2}_{\text{鼻子的位置}} \\ \underbrace{U_{j|3}}_{\text{由嘴巴推出的脸的位置}} = \underbrace{W_{3j}}_{\text{嘴巴和脸之间的空间关系}} \cdot \underbrace{u_3}_{\text{嘴巴的位置}} \end{array} \right. \end{array}$$

- 第二步：输入加权

$$\underbrace{C_{ij}}_{\text{权重}} \times \underbrace{U_{ji}}_{\text{高层特征}}$$

- 权重由动态路由（dynamic routing）确定
- 路由：神经网络把信息从底层VN传输到高层VN的活动

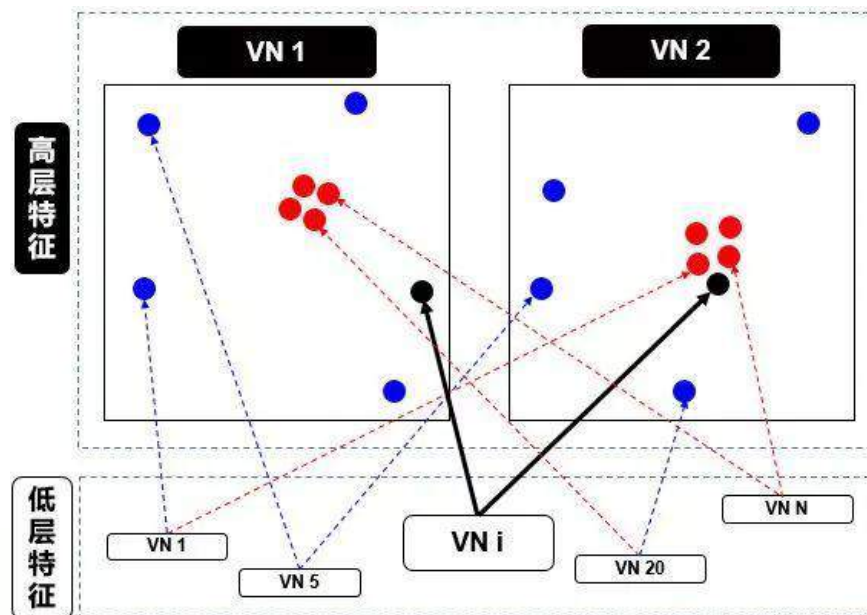


- 第三步：加权求和

$$s_j = \sum_i c_{ij} u_{j||i}$$

- 第四步：非线性激活

$$v_j = \underbrace{\frac{\|s_j\|}{1 + \|s_j\|^2}}_{\text{压扁}} \cdot \underbrace{\frac{s_j}{\|s_j\|}}_{\text{单位化}}$$



- squash函数:  $v_j$ 与 $s_j$ 同方向, 且 $v_j$ 的长度在0和1之间, 可以解释为第 $j$ 个VN具有给定特征的概率

- 动态路由：找出每一个“低层VNi”的输出最有可能贡献给哪个“高层VNj”，通过改变权重cij实现

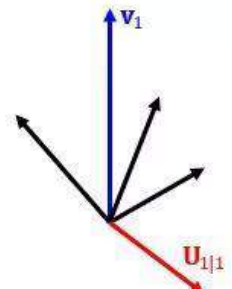
Procedure 1 Routing algorithm.

```

1: procedure ROUTING( $\hat{u}_{j|i}$ ,  $r$ ,  $l$ )
2:   for all capsule  $i$  in layer  $l$  and capsule  $j$  in layer  $(l + 1)$ :  $b_{ij} \leftarrow 0$ .
3:   for  $r$  iterations do
4:     for all capsule  $i$  in layer  $l$ :  $c_i \leftarrow \text{softmax}(\mathbf{b}_i)$  ▷ softmax computes Eq. 3
5:     for all capsule  $j$  in layer  $(l + 1)$ :  $\mathbf{s}_j \leftarrow \sum_i c_i \hat{\mathbf{u}}_{j|i}$ 
6:     for all capsule  $j$  in layer  $(l + 1)$ :  $\mathbf{v}_j \leftarrow \text{squash}(\mathbf{s}_j)$  ▷ squash computes Eq. 1
7:     for all capsule  $i$  in layer  $l$  and capsule  $j$  in layer  $(l + 1)$ :  $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i} \cdot \mathbf{v}_j$ 
   return  $\mathbf{v}_j$ 
    
```

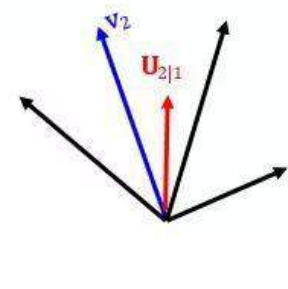
- 由 $U_{j|i}$  (“个人”预测)和 $\mathbf{v}_j$  (“共识”预测)的点积更新 $b_{ij}$

- $b_{ij} \uparrow$ , VNi的输出贡献给VNj的可能性 $\uparrow$
- $b_{ij} \downarrow$ , VNi的输出贡献给VNj的可能性 $\downarrow$



$$b_{11} = b_{11} + \underbrace{U_{1|1} \cdot \mathbf{v}_1}_{\text{很小一项}}$$

$$c_{11} = \frac{e^{b_{11}}}{e^{b_{11}} + e^{b_{12}}} \quad \downarrow$$



$$b_{12} = b_{12} + \underbrace{U_{2|1} \cdot \mathbf{v}_2}_{\text{很大一项}}$$

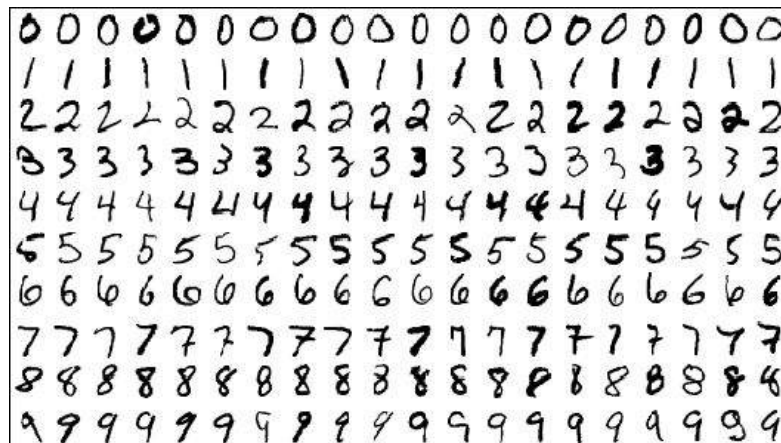
$$c_{12} = \frac{e^{b_{12}}}{e^{b_{11}} + e^{b_{12}}} \quad \uparrow$$



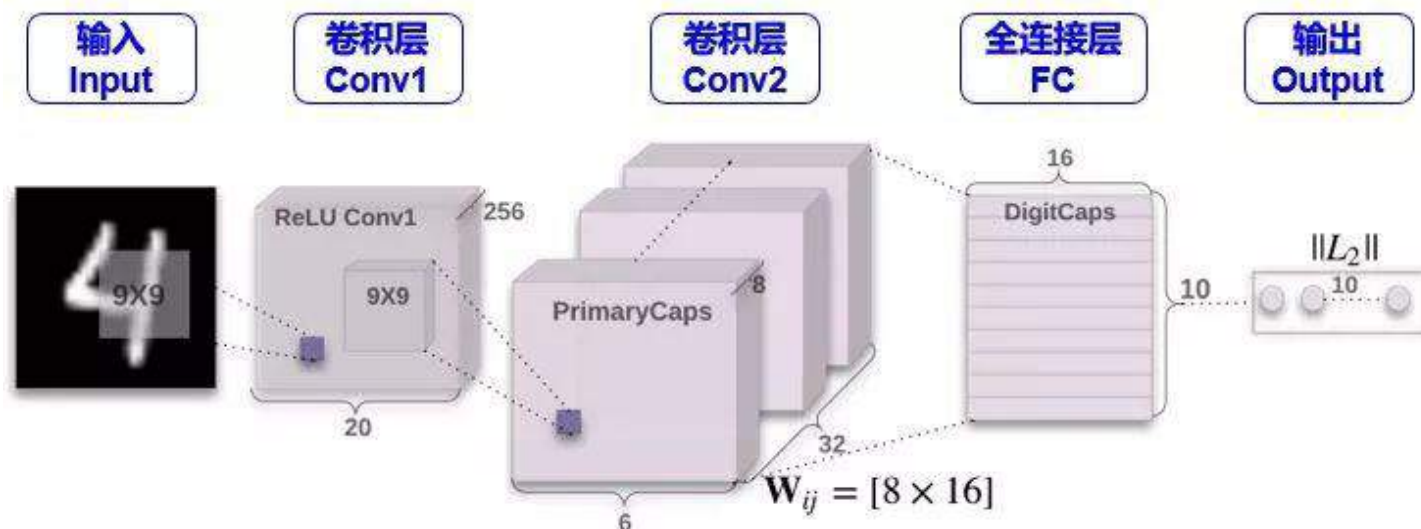


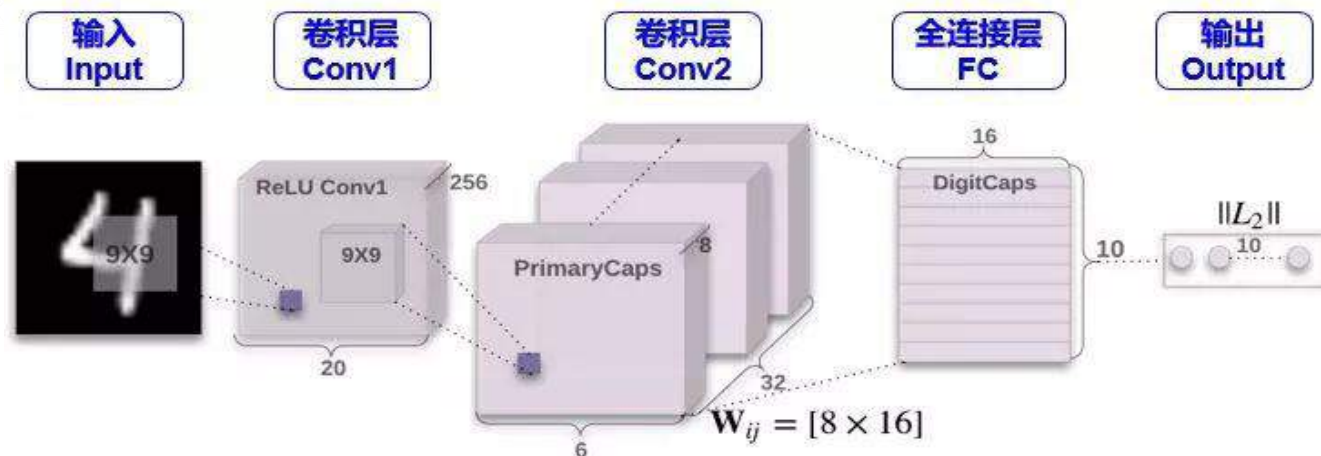
# 网络结构

- 以MNIST数据集为例，每幅图像为一个 $28 \times 28$ 像素的单元

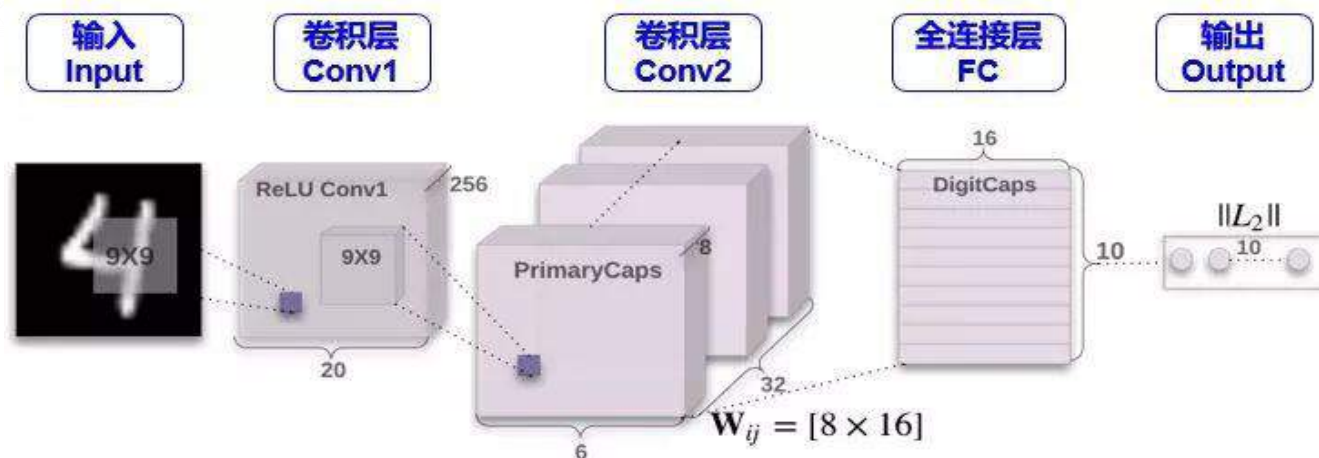


- CapsNet网络结构如下图：
  - 输入： $28 \times 28$ 的二维矩阵
  - 输出： $10 \times 1$ 的概率向量

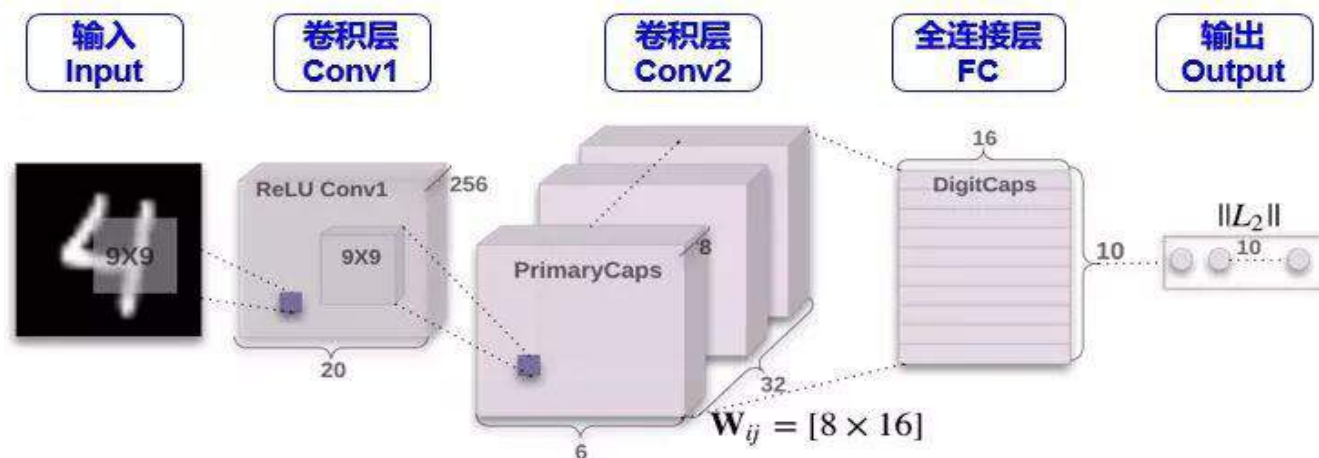




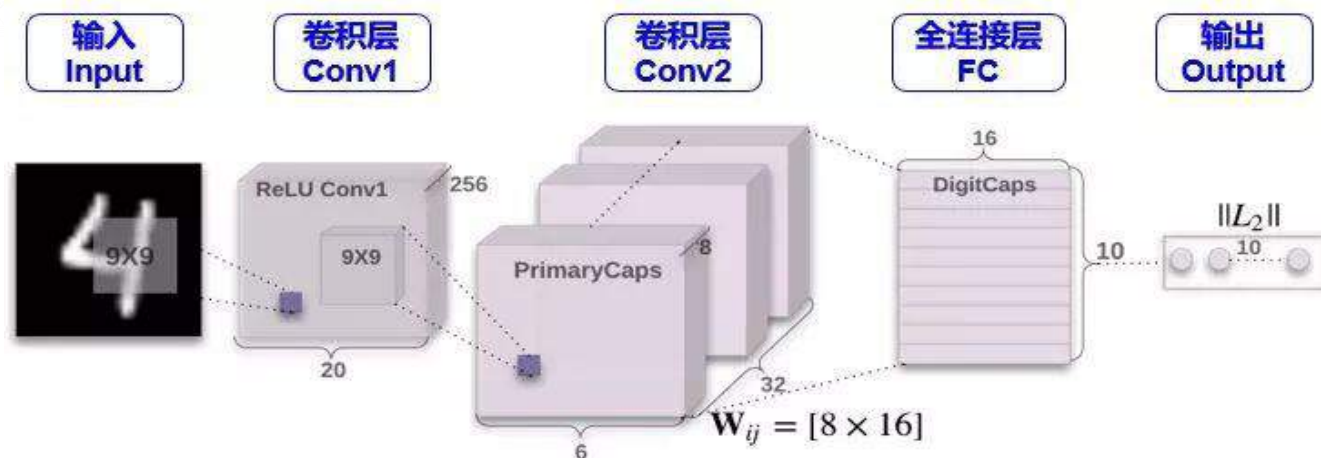
- 图像输入到低级特征 (Conv1)
  - 局部特征检测
  - 用256个stride为1的 $9 \times 9$ 的filter，得到一个 $20 \times 20 \times 256$ 的输出



- 低级特征到primary Capsule ( Conv2 )
  - 卷积胶囊层，用于储存低级别特征的向量
  - 32个stride为2的9×9的filter，每个通道有一个8维卷积胶囊



- primary Capsule到Digit Capsule (FC)
  - 存储高级别特征的向量
  - 为了让 $1 \times 8$ 和 $1 \times 16$ 的向量全连接，需要 $8 \times 16$ 的姿态矩阵
  - 动态路由
  - 输出10个 $v_j$



- Digit Capsule到输出 (output)
  - 分类
  - $v_j$ 的长度表示其表征的内容出现的概率
  - 输出的概率总和不等于1, 可以同时识别多个物体

- 损失函数

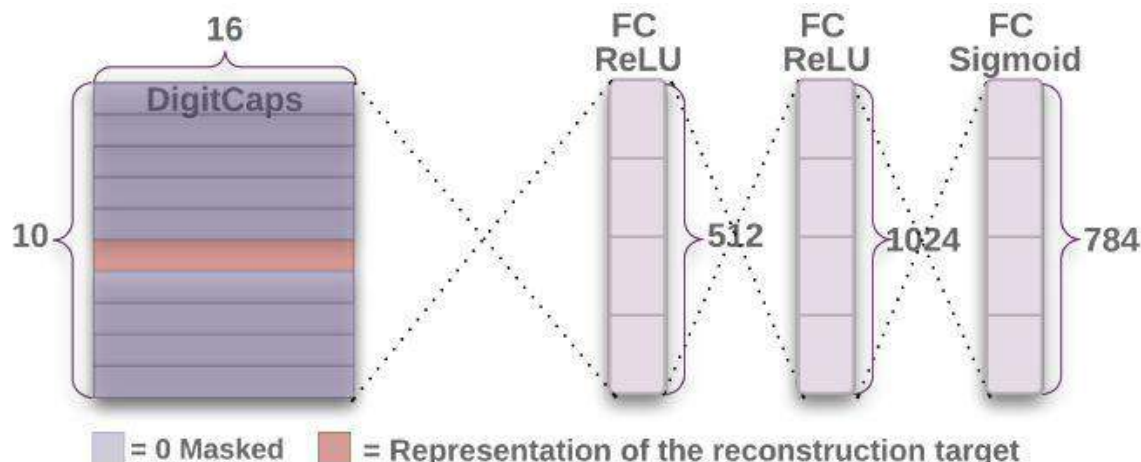
- 由于允许多个分类同时存在，所以不能直接使用交叉熵（cross-entropy）损失，而使用间隔损失（margin loss）

$$L_k = T_k \cdot \max(0, m^+ - \|\mathbf{v}_k\|)^2 + \lambda(1 - T_k) \cdot \max(0, \|\mathbf{v}_k\| - m^-)^2$$

- 其中

- $k$ 是分类
  - $T_k$ 是分类的指示函数（ $k$ 类存在为1，不存在为0）
  - $m^+$ 为上界，惩罚假阳性（false positive），即预测 $k$ 类存在但真实不存在
  - $m^-$ 为下界，惩罚假阴性（false negative），即预测 $k$ 类不存在但真实存在
  - $\lambda$ 是比例系数，调整两者比重
- 总的损失是各个样例损失之和

- 重构:



- 输出维度:  $784=28 \times 28$
- 重构损失 (reconstruction loss): 把最终输出和最初输入的784个单元上的像素值相减并求平方和。
- 总体损失 (total loss) = 间隔损失 +  $\alpha \cdot$  重构损失
  - $\alpha=0.005$





# 优劣分析

- 优点

- 相比于CNN，提高了对图像变换的健壮性，在图像分割中也表现出色
- 需要的训练数据更少
- 动态路由使得可以处理更加复杂的场景

- 缺点

- 胶囊拥挤，如果一个胶囊网络彼此之间太过接近，则无法检测到同一类型的两个对象
- 胶囊网络很慢，由于内部循环的路由协议算法



# 应用总结

- 人脸识别、物体检测
- 图片信息推理，图片描述生成
- 文本挖掘



## 参考文献

- [1] Sabour S, Frosst N, Hinton G E. Dynamic routing between capsules[C]//Advances in Neural Information Processing Systems. 2017: 3859–3869.
- [2]<http://www.dataguru.cn/article-12895-1.html> 胶囊(向量神经)网络
- [3]<https://www.leiphone.com/news/201710/seYRjGDt30yXcNSr.html> 胶囊间的动态路由
- [4]知乎“如何看待Hinton的论文Dynamic Routing Between Capsules”. 云梦局客



大成若缺，其用不弊。  
大盈若冲，其用不穷。  
大直若屈。大巧若拙。  
大辩若讷。静胜躁，寒  
胜热。清静为天下正。

# 谢谢！

