度量学习

度量学习（Metric Learning）它的目的就是用来比较两个对象之间的相似程度，通常用于图像识别领域，比如人脸的相似性、姿势的相似性等。相似性的度量函数d(x,y)一般根据处理的问题进行选择，但是要尽量满足以下的条件：

1. $d(x,x)=0$，表示向量x到自身的距离为0
2. $d(x,x)\geq 0$，非负性
3. $d(x,y)=d(y,x)$，对称性
4. $d(x,k)+d(k,y)\geq d(x,y)$，三角不等式

所以常见的相似性度量方法有：欧氏距离、曼哈顿距离、切比雪夫距离、闵可夫斯基距离、马氏距离、夹角余弦、信息熵等。相似性度量描述链各个对象之间的相似程度，假设特征向量 z和y分别表示为：

$$x=\left(x_1,x_2,\cdots,x_n\right)$$

$$y=\left(y_1,y_2,\cdots,y_n\right)$$

相似性度量函数 d(x,y) 越小则表示 x 和 y 越相似。距离度量可以理解为相似度， 任何方法只要用到相似度，就会涉及到度量学习。度量学习的目的就是使得相似的样本距离更小，不相似的样本距离更大。

常见的度量学习损失函数

这里一图像领域行人识别进行举例

1. 输入为两张图片，每一对训练图片都有一个标签 y,y=1 表示两张图片属于同一个人（正样本对），反之y=0，表示他们属于不同的人（负样本对）；则它的对比损失函数为：

$$L_c=yd^2_{I_a,I_b}+(1-y)(\alpha-d_{I_a,I_b})^2_+$$

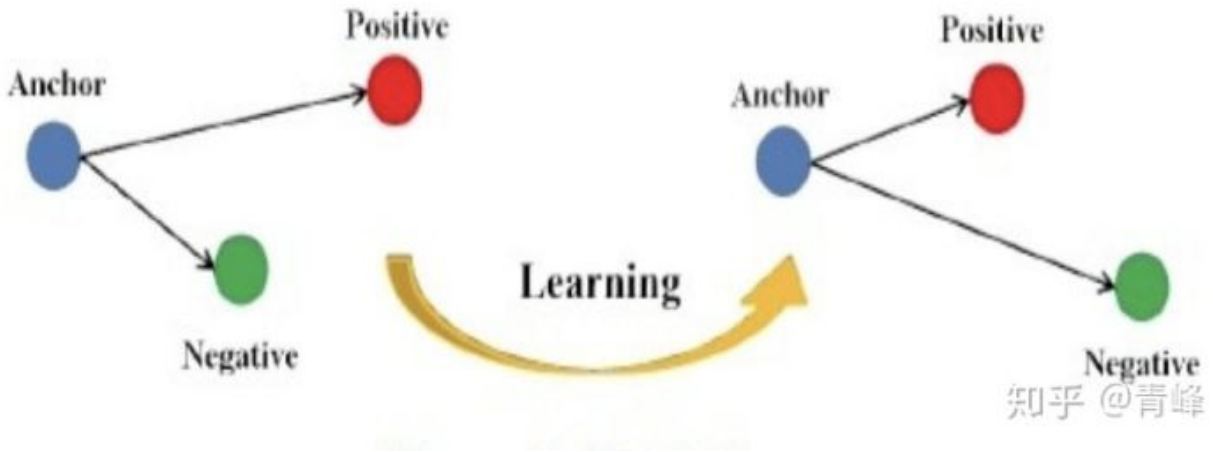
当输入为正样本对的时候，$d(I_a,I_b)$ 逐渐减小，属于同一个人的图片会持续在特征空间中形成聚类；反之，当输入的是负样本对时，$d(I_a,I_b)$ 逐渐变大，知道超过设定的阈值（安全距离） α 。通过最小化损失函数，最后可以使正样本对之间距离逐渐变小，负样本对之间距离逐渐变大，从而满足行人重识别任务的需要。

   - 三元组损失（Triplet loss）

三元组损失是一种被广泛应用的度量学习损失，之后的大量度量学习方法也是基于三元组损失演变而来。顾名思义，三元组损失需要三张输入图片。和对比损失不同，一个输入的三元组（Triplet）包括一对正样本对和一对负样本对。三张图片分别命名为固定图片(Anchor) a、正样本图片(Positive)p和负样本图片(Negative) n 。图片 a 和图片 p 为一对正样本对，图片 a 和图片 n 为一对负样本对。则三元组损失表示为：

$$L_t = (d_{a,p} - d_{a,n} + \alpha)_+$$

如下图所示，三元组可以拉近正样本对之间的距离，推开负样本对之间的距离，最后使得相同ID的行人图片在特征空间里形成聚类，达到行人重识别的目的。如下图所示：



改进：原版的Triplet loss只考虑正负样本对之间的相对距离，而并没有考虑正样本对之间的绝对距离，为此提出改进三元组损失(Improved triplet loss)：

$$L_i t = d_{a,p} + (d_{a,p} - d_{a,n} + \alpha)_+$$

保证网络不仅能够在特征空间把正负样本推开，也能保证正样本对之间的距离很近。

- 四元组损失(Quadruplet loss)
- 难样本采样三元组损失(Triplet loss with batch hard mining, TriHard loss)
- 边界挖掘损失(Margin sample mining loss, MSML)

这里不再详细介绍。

度量学习算法通过学习生成什么样的距离度量来帮助数据间的重要关系。

度量学习中的标签以成对约束的形式指定,包括已知相似的集合S={(xi,xj)}S={(xi,xj)}，以及不相似的集合对D={xi,xj}D={xi,xj}。 原始的Metric Learning方法尝试学习马氏距离度量

$$d_A(x_i, x_j) = \sqrt{(x_i - x_j)^T A (x_i - x_j)}\ ,$$

where $A \in \mathbb{R}^{m \times m}$ is a positive semi-definite matrix [51]. This is equivalent to projecting each input x to a new space $\mathbb{R}^m$ in which their euclidean distances obey the desired constraints. There are many different ways to create such a metric. The most original approach was to globally solve the following convex optimization problem:

$$\min_A \sum_{(x_i,x_j)\in\mathcal{S}} d_A(x_i, x_j)^2$$

$$\text{s.t.} \ \sum_{(x_i,x_j)\in\mathcal{D}} d_A(x_i, x_j)^2 \geq 1 \text{ and } A \succeq 0.$$

一个输入x，我们把离x近的数据点作为它的target neighbors。并且假设x的target neighbors构建了不同类标之间的边界。不同类标的输入侵入边界被作为importors。这里，学习的目标就是最小化importors的个数。

一种为人知模型是LMNN，它使用2个损失函数来表达上述目标函数。

- Pull Loss，公式如下：

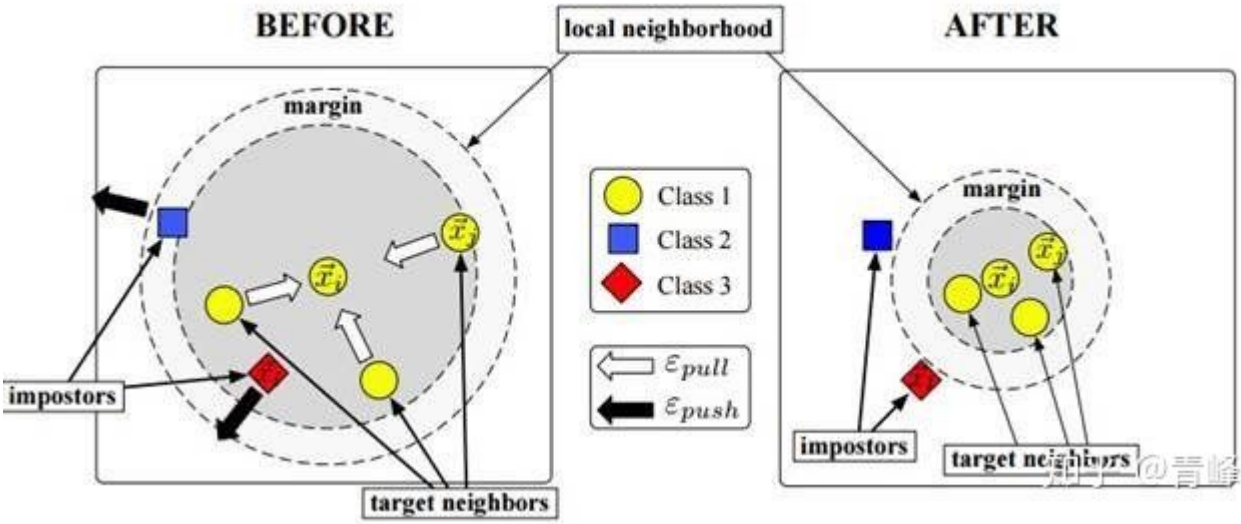
$$\mathcal{L}_{pull}(d) = \sum_{j \rightsquigarrow i} d(x_i, x_j)^2,$$

其中，j 是 i 的target neighbor。

- Push Loss，公式如下：

$$\mathcal{L}_{push}(d) = \sum_{i,j \rightsquigarrow i} \sum_{k} (1 - y_{ik})[1 + d(x_i, x_j)^2 - d(x_i, x_k)^2]_+$$

其中，如果i和k拥有同一个类标，则 $y_{ik} = 1$,否则， $y_{ik} = 0$ ， $[z]+ = max(z, 0)$ ，完整的损失函数是结合上述两个损失函数。


BEFORE
local neighborhood
AFTER
margin
Class 1
Class 2
Class 3
ε_{pull}
ε_{push}
impostors
target neighbors
margin
impostors
target neighbors