# 算法原理

我们队本次比赛使用的目标检测算法是YOLO3，目标追踪算法为DeepSort。
**Person Detection (YOLO3) - > Person Tracking (DeepSort)**
通过行人识别算法检测行人目标，将行人目标特征及轨迹传入追踪算法，实现行人特征预测及轨迹预测和状态估计。

## 目标检测部分 - YOLO3:

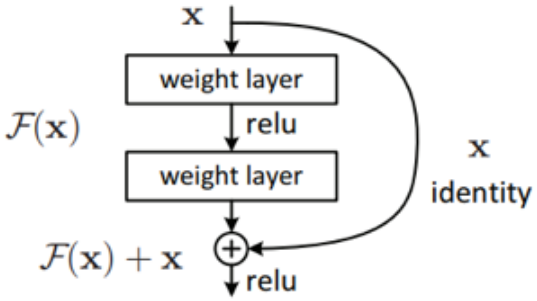
YOLO3 主要的改进有三点：1.调整了网络结构；2.利用多尺度特征进行对象检测；3.对象分类用Logistic取代了softmax。
在几前几个版本中保持了速度优势的同时，提升了预测精度，尤其是加强了对小目标的识别

### 网络结构

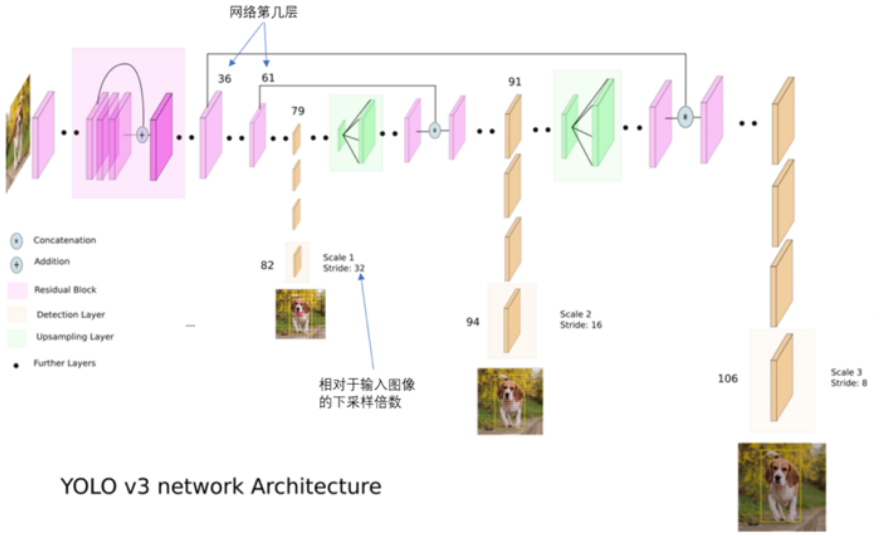
在基本的图像特征提取方面，YOLO3采用了称之为 Darknet-53 的网络结构（含有53个卷积层），它借鉴了残差网络 residual network 的做法，在一些层之间设置了快捷链路（shortcut connections）

| | Type | Filters | Size | | Output | |
|---|---|---|---|---|---|---|
| | Convolutional | 32 | 3 | 3 | 256 | 256 |
| | Convolutional | 64 | 3 | 3 / 2 | 128 | 128 |
| 1 | Convolutional | 32 | 1 | 1 | | |
| | Convolutional | 64 | 3 | 3 | | |
| | Residual | | | | 128 | 128 |
| | Convolutional | 128 | 3 | 3 / 2 | 64 | 64 |
| 2 | Convolutional | 64 | 1 | 1 | | |
| | Convolutional | 128 | 3 | 3 | | |
| | Residual | | | | 64 | 64 |
| | Convolutional | 256 | 3 | 3 / 2 | 32 | 32 |
| 8 | Convolutional | 128 | 1 | 1 | | |
| | Convolutional | 256 | 3 | 3 | | |
| | Residual | | | | 32 | 32 |
| | Convolutional | 512 | 3 | 3 / 2 | 16 | 16 |
| 8 | Convolutional | 256 | 1 | 1 | | |
| | Convolutional | 512 | 3 | 3 | | |
| | Residual | | | | 16 | 16 |
| | Convolutional | 1024 | 3 | 3 / 2 | 8 | 8 |
| 4 | Convolutional | 512 | 1 | 1 | | |
| | Convolutional | 1024 | 3 | 3 | | |
| | Residual | | | | 8 | 8 |
| | Avgpool | | Global | | | |
| | Connected | | 1000 | | | |
| | Softmax | | | | | |

上图的Darknet-53网络采用256*256*3作为输入，最左侧那一列的1、2、8等数字表示多少个重复的残差组件。
每个残差组件有两个卷积层和一个快捷链路，示意图如下：



## 利用多尺度特征进行对象检测



YOLO v3 network Architecture

YOLO2曾采用passthrough结构来检测细粒度特征，在YOLO3更进一步采用了3个不同尺度的特征图来进行对象检测。

卷积网络在79层后，经过下方几个黄色的卷积层得到一种尺度的检测结果。相比输入图像，这里用于检测的特征图有32倍的下采样。比如输入是416*416的话，这里的特征图就是13*13了。由于下采样倍数高，这里特征图的感受野比较大，因此适合检测图像中尺寸比较大的对象。

为了实现细粒度的检测，第79层的特征图又开始作上采样（从79层往右开始上采样卷积），然后与第61层特征图融合（Concatenation），这样得到第91层较细粒度的特征图，同样经过几个卷积层后得到相对输入图像16倍下采样的特征图。它具有中等尺度的感受野，适合检测中等尺度的对象。

最后，第91层特征图再次上采样，并与第36层特征图融合（Concatenation），最后得到相对输入图像8倍下采样的特征图。它的感受野最小，适合检测小尺寸的对象。

9种尺度的先验框随着输出的特征图的数量和尺度的变化，先验框的尺寸也需要相应的调整。

| 特征图 | 13*13 | | | 26*26 | | | 52*52 | | |
|---|---|---|---|---|---|---|---|---|---|
| 感受野 | 大 | | | 中 | | | 小 | | |
| 先验框 | (116x90) | (156x198) | (373x326) | (30x61) | (62x45) | (59x119) | (10x13) | (16x30) | (33x23) |

YOLO3延续了K-means聚类得到先验框的尺寸这种方法，为每种下采样尺度设定3种先验框，总共聚类出9种尺寸的先验框。在COCO数据集这9个先验框是：(10x13)，(16x30)，(33x23)，(30x61)，(62x45)，(59x119)，(116x90)，(156x198)，(373x326)。

分配上，在最小的13*13特征图上（有最大的感受野）应用较大的先验框(116x90)，(156x198)，(373x326)，适合检测较大的对象。中等的26*26特征图上（中等感受野）应用中等的先验框(30x61)，(62x45)，(59x119)，适合检测中等大小的对象。较大的52*52特征图上（较小的感受野）应用较小的先验框(10x13)，(16x30)，(33x23)，适合检测较小的对象。

### 对象分类softmax改成logistic

预测对象类别时不使用softmax，改成使用logistic的输出进行预测，支持目标细分属性识别（比如区分哈士奇和卷毛）

## 目标追踪部分- DeepSort:

DeepSort (simple online and realtime tracking with a deep association metric)是在Sort(simple online and realtime tracking)目标追踪基础上的改进。在Sort的基础上引入了深度学习行人 ReID 模型，在每一帧的目标追踪过程中，提取每一帧目标的特征对并对下一帧进行相似度匹配，解决了遮挡和表征突变的情况下目标丢失的问题，算法的核心是使用了递归的卡尔曼滤波和逐帧的特征数据关联。

### 1.轨迹处理和状态估计

运动状态估计：
通过8维的状态空间 (u,v,γ,h,$x^{\cdot}$,$y^{\cdot}$,$\gamma^{\cdot}$,$h^{\cdot}$)，来进行运动状态的描述，其中(u,v)是bounding box的中心坐标，r是长宽比，h表示高度。其余四个变量表示对应的在图像坐标系中的速度信息。使用一个基于匀速运动模型（ constant velocity motion）和线性观测模型（ linear observation ）的标准kalman滤波器进行目标运动状态的预测，预测的结果为(u,v,r,h)，作为直接观察对象的状态。

轨迹处理:
首先对于每条轨迹都有一个阈值a用于记录轨迹从上一次成功匹配到前时刻的时间。当该值大于提前设定的阈值则认为改轨迹终止，直观上说就是长时间匹配不上的轨迹认为已经结束。然后在匹配时，对于没有匹配成功的detections都认为可能产生新的轨迹。但由于这些detections可能是一些false alarms，所以对这种情形新生成的轨迹标注状态'tentative' ,然后观查在接下来的连续若干帧（论文中是3帧）中是否连续匹配成功，是的话则认为是新轨迹产生，标注为'confirmed',否则则认为是假性轨迹,状态标注为'deleted'。

### 2. 指派问题

- 使用平方马氏距离来度量预测track的Kalman状态（bbox的几何位置）和新到来detection之间的距离；

$$d^{(1)}(i,j) = (\boldsymbol{d}_j - \boldsymbol{y}_i)^{\mathrm{T}} \boldsymbol{S}_i^{-1} (\boldsymbol{d}_j - \boldsymbol{y}_i), \qquad (1)$$

- 使用cosine距离来度量各个track的appearance feature（128维）和detection feature之间的距离，来跟准确地预测ID；

$$d^{(2)}(i,j) = \min\{1 - \boldsymbol{r}_j^{\mathrm{T}} \boldsymbol{r}_k^{(i)} \mid \boldsymbol{r}_k^{(i)} \in \mathcal{R}_i\}. \qquad (3)$$

- 引入两个二值函数来限制assignment矩阵，分别比较平方马氏距离以及cosine距离和阈值的大小来进行判断，将两个函数结合起来对矩阵进行限制；

$$b_{i,j}^{(1)} = \mathbb{1}[d^{(1)}(i,j) \leq t^{(1)}]$$

$$b_{i,j} = \prod_{m=1}^{2} b_{i,j}^{(m)}. \quad t^{(2)}]$$

- 使用 **combined** 距离来作为 **cost matrix**进行度量各个**track**和**detection**之间的距离，这里文中只使用**cosine**距离进行度量（即将**lambda**设置为**0**），使用马氏距离排除不可能的情况，既基于由卡尔曼滤波器推断的可能的物体位置忽略不可行的分配。

$$c_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^{(2)}(i,j)$$

## 3.级联匹配

- 物体被遮挡一段时间后，卡尔曼滤波预测的不确定性大大增加并且状态空间上可观察性变得很低，并且马氏距离更倾向于不确定性更大的track，这是由于减少了detection的标准差距预计的轨迹的距离。因此这里引入级联匹配，优先匹配detection与最近出现的track；
- Matching Cascade算法见下，在最后结束算法时使用SORT中的IOU距离来解决局部遮挡的问题，通过计算unmatched tracks（只有前一帧是unmatched的） 和 unmatched detection的IOU distance。

在匹配的最后阶段还对unconfirmed和age=1的未匹配轨迹进行基于IoU的匹配，这样可以缓解因为特征角度突变或者部分遮挡导致的较大差异。

## 4. Appearance 描述

通过在大规模re-id数据集上pre-trained深度网络来提取128维的appearance特征，用L2正则化将特征投影到单位超球面上来与余弦距离进行匹配，网络结构见下（需要注意的是此处的detection用的是POI中的detection坐标，文中的CNN网络用于提取bbox中的特征）。

| Name | Patch Size/Stride | Output Size |
|---|---|---|
| Conv 1 | $3 \times 3/1$ | $32 \times 128 \times 64$ |
| Conv 2 | $3 \times 3/1$ | $32 \times 128 \times 64$ |
| Max Pool 3 | $3 \times 3/2$ | $32 \times 64 \times 32$ |
| Residual 4 | $3 \times 3/1$ | $32 \times 64 \times 32$ |
| Residual 5 | $3 \times 3/1$ | $32 \times 64 \times 32$ |
| Residual 6 | $3 \times 3/2$ | $64 \times 32 \times 16$ |
| Residual 7 | $3 \times 3/1$ | $64 \times 32 \times 16$ |
| Residual 8 | $3 \times 3/2$ | $128 \times 16 \times 8$ |
| Residual 9 | $3 \times 3/1$ | $128 \times 16 \times 8$ |
| Dense 10 | | 128 |
| Batch and $\ell_2$ normalization | | 128 |

**Table 1**: Overview of the CNN architecture. The final batch and $\ell_2$ normalization projects features onto the unit hypersphere.