山东大学
SHANDONG UNIVERSITY

信息基础2 机器学习与深度学习

实验五

陈雷

山东大学信息科学与工程学院

实验内容

实验一：常规神经网络函数逼近实验

实验二：基于LeNet-5的MNIST字符识别

实验三：ResNet-18

实验四：Selective search

实验五：Yolo

期末项目：命题项目和自选项目

实验五：Yolo

<<You Only Look Once: Unified, Real-Time Object Detection>>是2016年CVPR上发表的一篇目标检测的文章，从题目就可以看出YOLO的两个特点：

unified：single-stage检测算法。

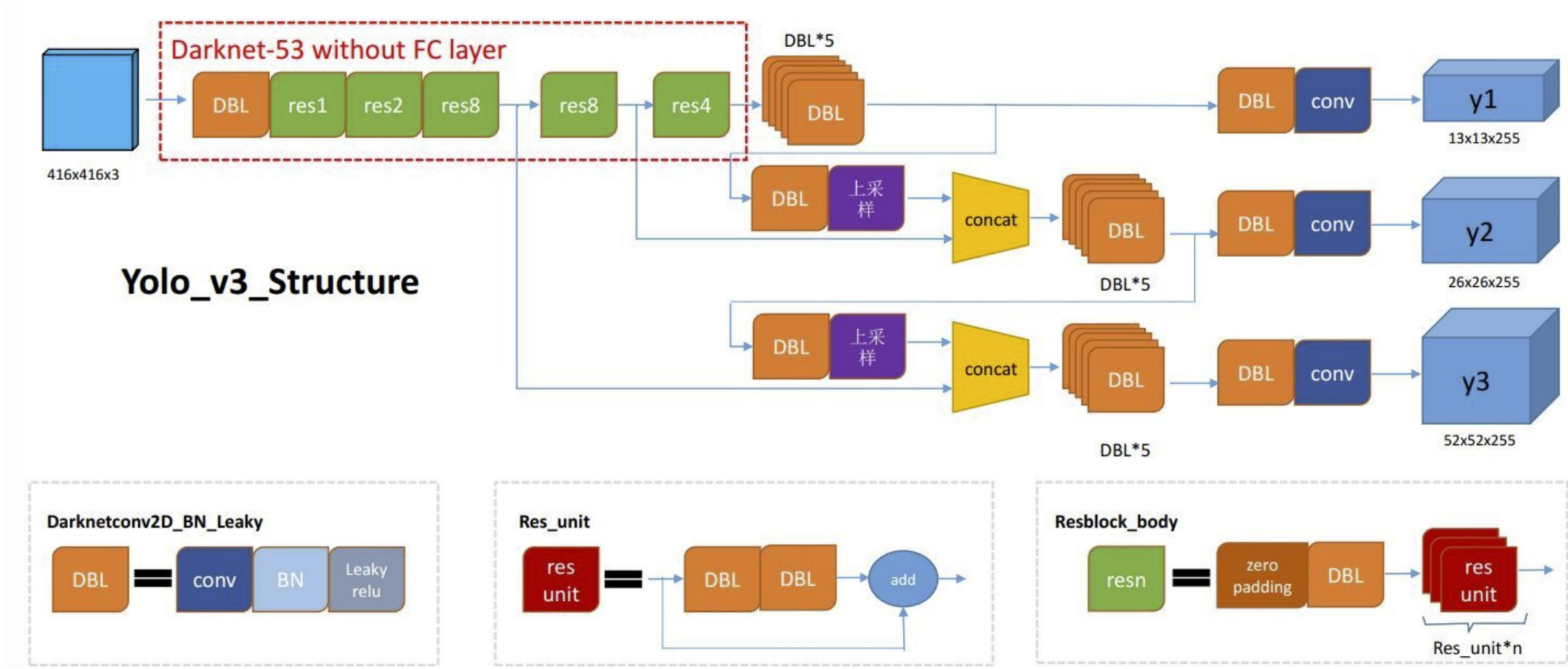
real-time：实时检测，速度很快。

实验五：Yolo

Yolo v3作为Yolo系列的经典算法，对Yolo v1和Yolo v2既有保留又有改进：

1. 从Yolo v1开始，Yolo算法就是通过划分单元格来做检测，只是划分的数量不一样。
2. 采用“leaky ReLU”作为激活函数。端到端进行训练，只需关注输入端和输出端。
3. 从Yolo v2开始，Yolo就用batch normalization（BN）作为正则化、加速收敛和避免过拟合的方法，把BN层和leaky ReLU层接到每一层卷积层之后。
4. 多尺度训练。在速度和准确率之间tradeoff。

实验五：Yolo


Darknet-53 without FC layer
DBL
res1
res2
res8
res8
res4
DBL*5
DBL
416x416x3
DBL
conv
y1
13x13x255
DBL
上采样
concat
DBL
DBL*5
DBL
conv
y2
26x26x255
DBL
上采样
concat
DBL
DBL*5
DBL
conv
y3
52x52x255
Yolo_v3_Structure
Darknetconv2D_BN_Leaky
DBL
conv
BN
Leaky relu
Res_unit
res unit
DBL
DBL
add
Resblock_body
resn
zero padding
DBL
res unit
Res_unit*n

实验五：Yolo

DBL: 是Yolo v3的基本组件，具体为Darknetconv2D+BN+Leaky ReLU。对于v3来说，BN和Leaky ReLU与卷积层不可分离（最后一层卷积除外），共同构成了最小组件。

resn： n代表数字，表示这个res_block里含有多少个res_unit，是Yolo v3的大组件。借鉴了ResNet的残差结构，使用这种结构可以让网络结构更深（从v2的darknet-19上升到v3的darknet-53，前者没有残差结构）。res_block的基本组件也是DBL。

concat： 张量拼接。将darknet中间层和后面某一层的上采样进行拼接。

实验五：Yolo

整个Yolo v3主体包含252层，组成如下：

Total: 252

BatchNormalization: 72

Concatenate: 2

Conv2D: 75

InputLayer: 1

LeakyReLU: 72

UpSampling2D: 2

ZeroPadding2D: 5

实验五：Yolo

整个Yolo v3结构里面，没有池化层和全连接层。

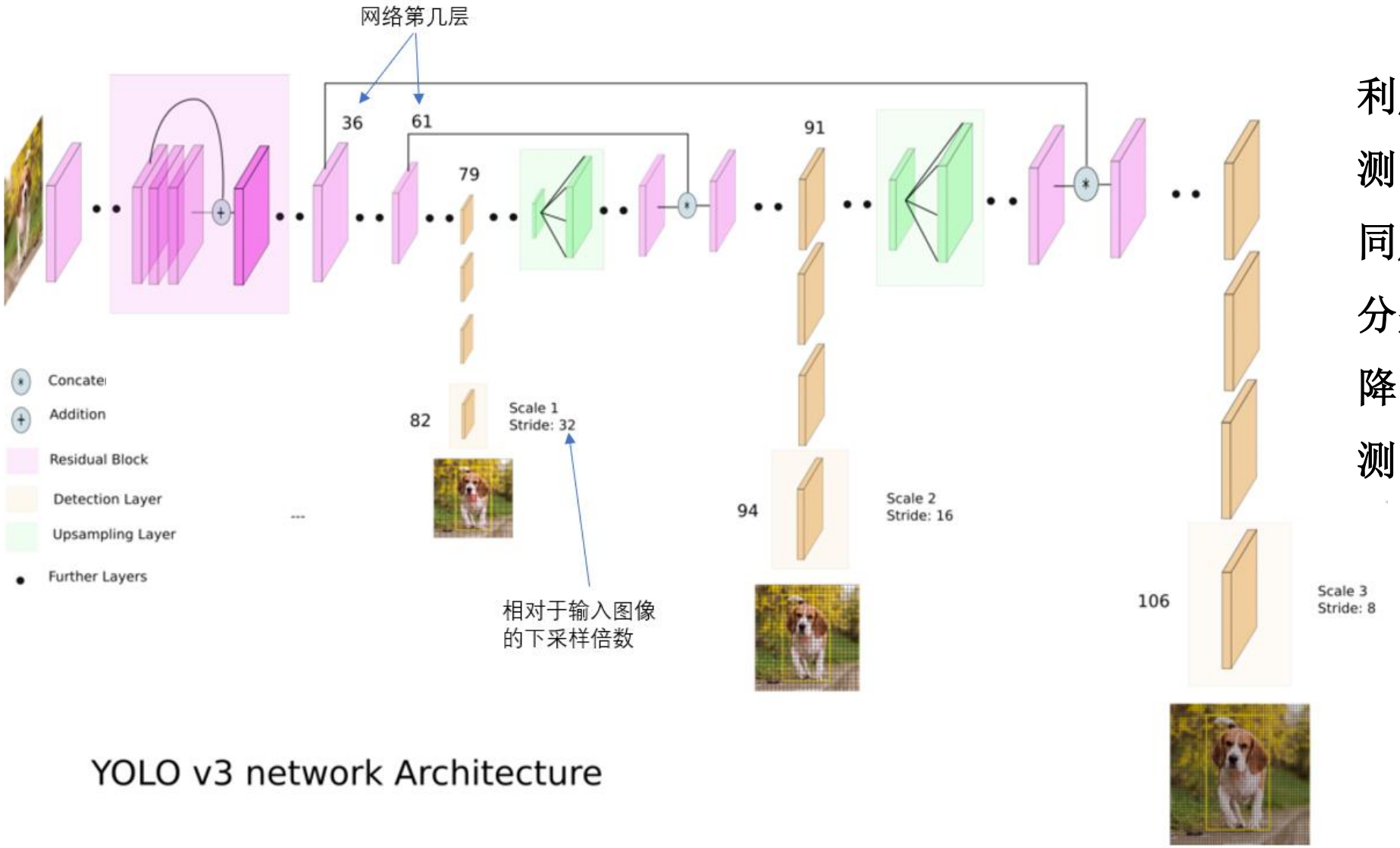
Darknet-53采用了ResNet这种跳层连接方式，Yolo_v3使用了darknet-53前面的52层。

前向传播过程中，张量的尺寸变换是通过改变卷积核的步长来实现的。要经历5次缩小，会将特征图缩小到原输入尺寸的1/32。

	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			

Table 1. **Darknet-53.**

实验五：Yolo



YOLO v3 network Architecture

利用多尺度特征进行检测，Yolo v3输出了3个不同尺度的feature map，分别是在32倍降、16倍降、8倍降采样时进行检测。

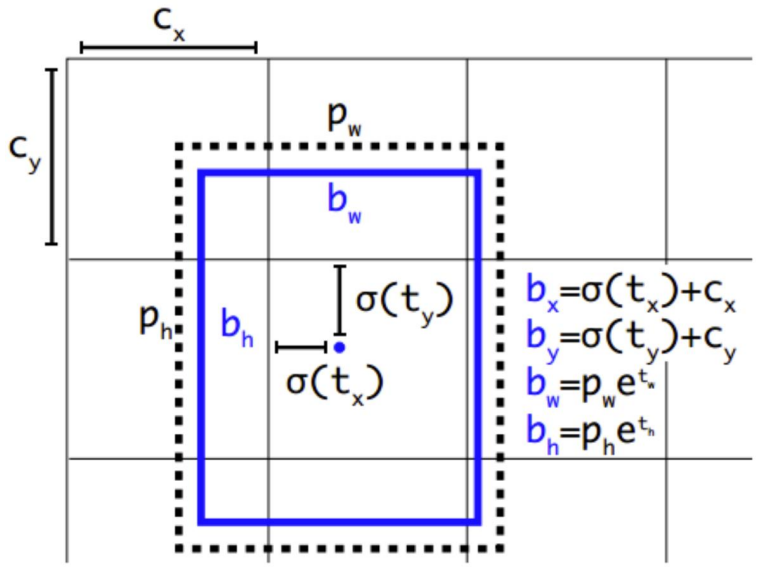
实验五：Yolo

feature map中的每一个cell都会预测3个边界框（bounding box），每个bounding box都会预测：

（1）每个框的位置（4个值，中心坐标t_x和t_y，框的高度b_h和宽度b_w）

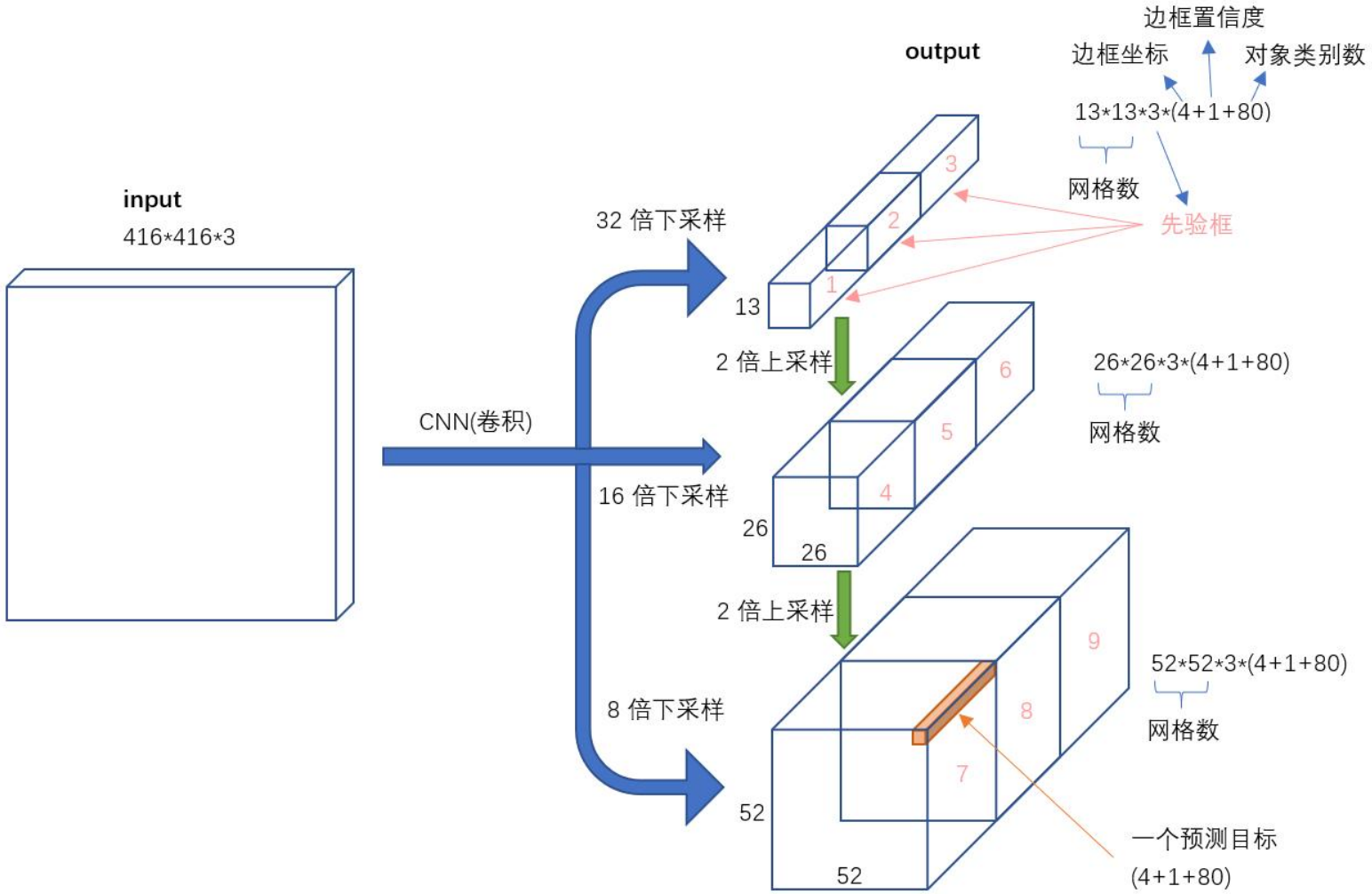
（2）一个objectness prediction

（3）N个类别，coco数据集80类，voc20类。



实验五：Yolo

Yolo v3输入到输出映射

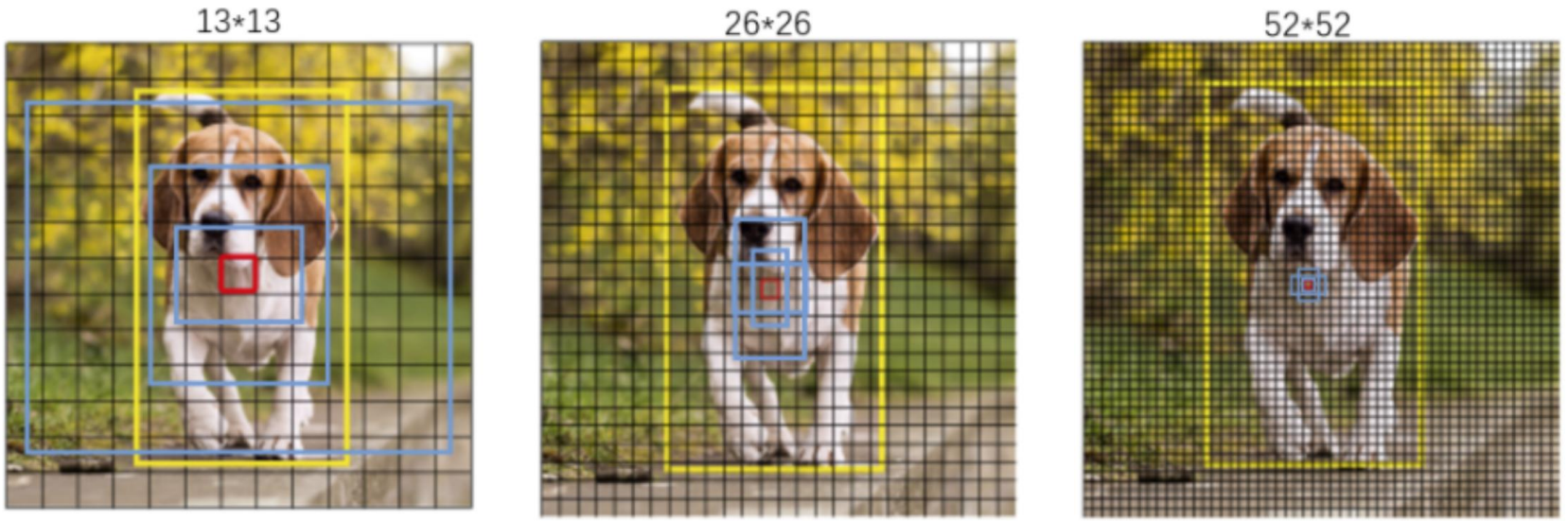


实验五：Yolo

三次检测，每次对应的感受野不同，32倍降采样的感受野最大，适合检测大的目标。在输入为416×416时，

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)

9种先验框的尺寸，下图中蓝色框为聚类得到的先验框。黄色框是ground truth，红色框是对象中心点所在的网格。



实验五：Yolo

Yolo v4在Yolo v3的基础上作了以下改进：

1. 输入端采用mosaic数据增强。

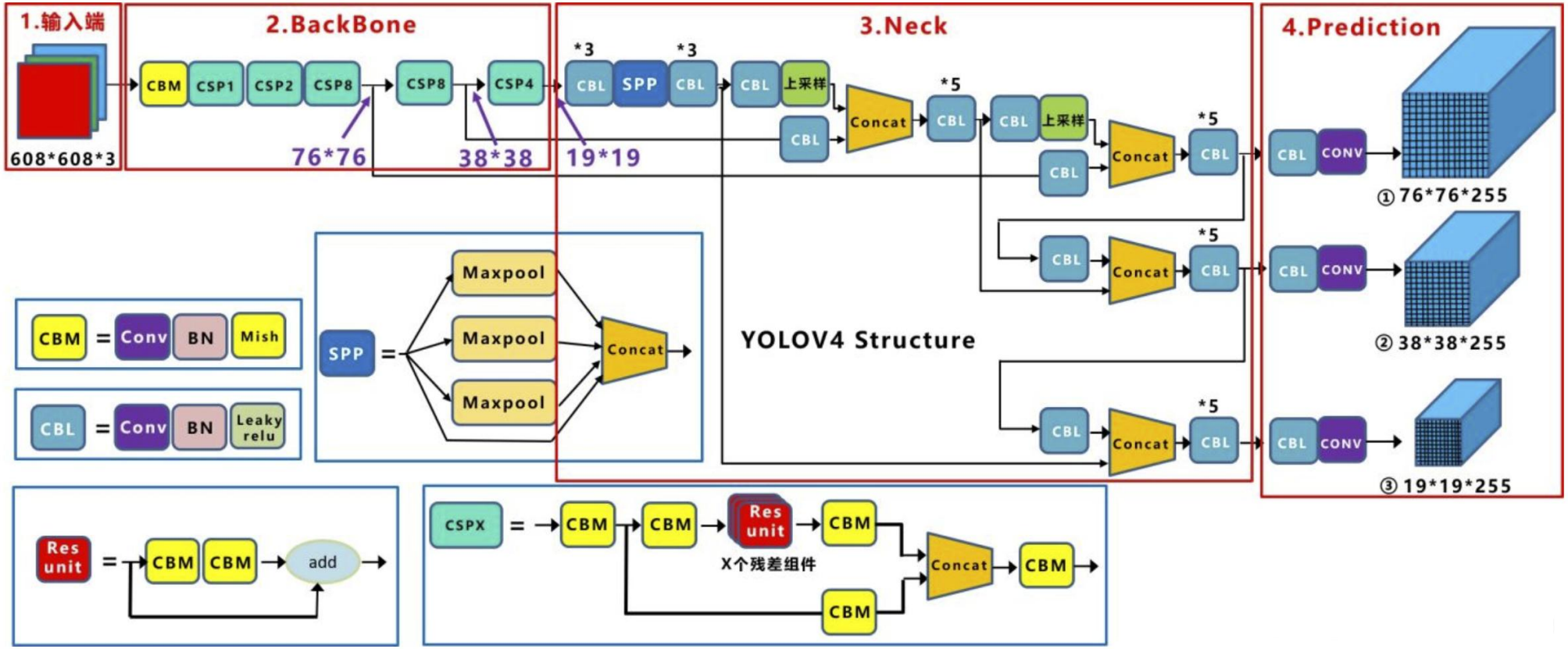
2. Backbone上采用了CSPDarknet53、Mish激活函数、Dropblock等方式。（cspnet减少了计算量的同时可以保证准确率）。

Mish函数为 $Mish = x * tanh(ln(1 + e^x))$

3. Neck中采用了SPP、FPN+PAN的结构。

4. 输出端采用CIOU_Loss、DIOU_nms操作。

实验五：Yolo


1.输入端
608*608*3
2.BackBone
CBM
CSP1
CSP2
CSP8
CSP8
CSP4
76*76
38*38
19*19
3.Neck
*3
CBL
SPP
CBL
*3
CBL
上采样
CBL
Concat
*5
CBL
CBL
上采样
CBL
Concat
*5
CBL
CBL
Concat
*5
CBL
CBL
Concat
*5
CBL
YOLOV4 Structure
4.Prediction
CBL
CONV
①76*76*255
CBL
CONV
②38*38*255
CBL
CONV
③19*19*255
CBM = Conv BN Mish
CBL = Conv BN Leaky relu
SPP = Maxpool Maxpool Maxpool Concat
Res unit = CBM CBM add
CSPX = CBM CBM Res unit CBM Concat CBM CBM
X个残差组件

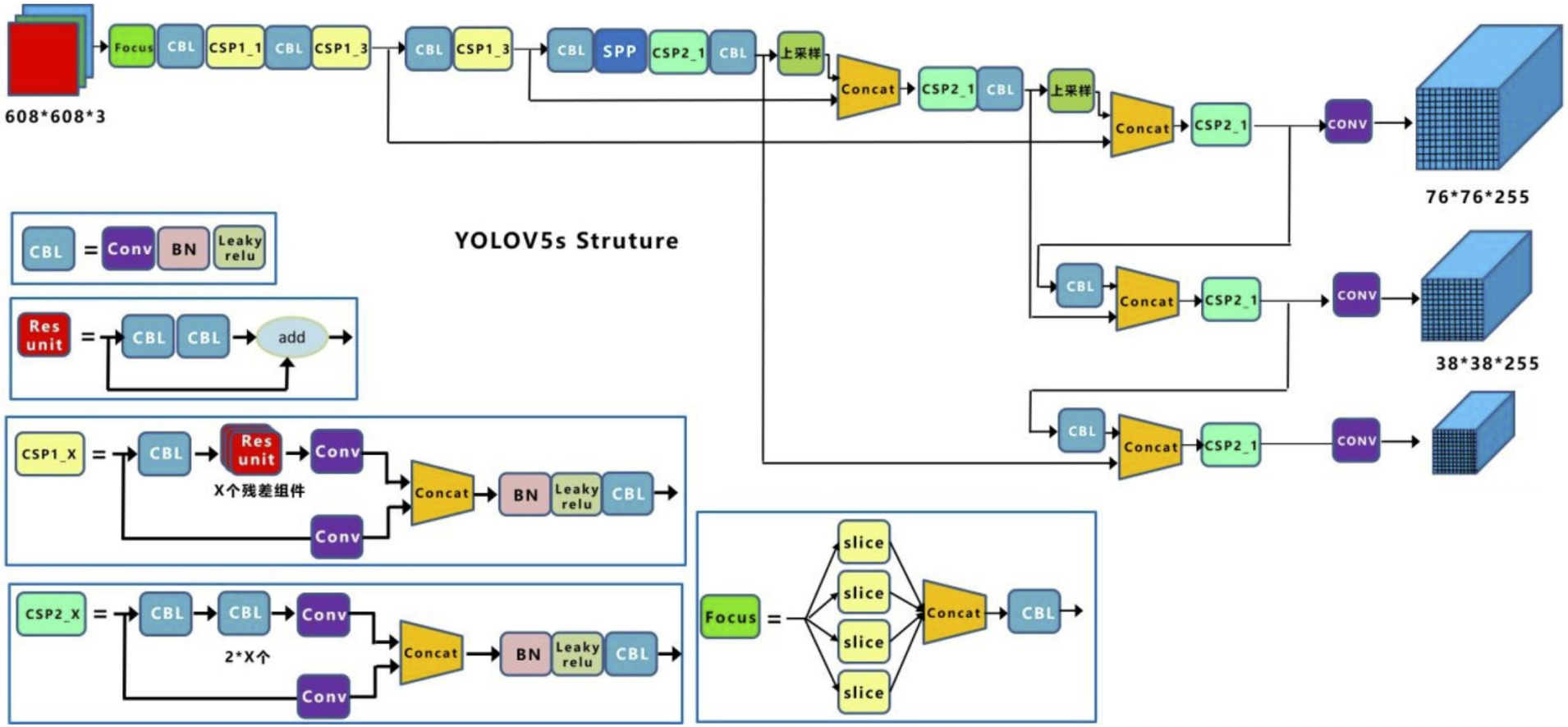

山东大学
1901
SHANDONG UNIVERSITY

实验五：Yolo

Yolo v5的输入端采用了和Yolo v4一样的Mosaic数据增强的方式。随机缩放、随机裁剪、随机排布的方式进行拼接，对于小目标的检测效果很不错。在Yolo算法中，针对不同的数据集，都会有初始设定长宽的锚框。

在网络训练中，网络在初始锚框的基础上输出预测框，进而和真实框groundtruth进行比对，计算两者差距，再反向更新，迭代网络参数。

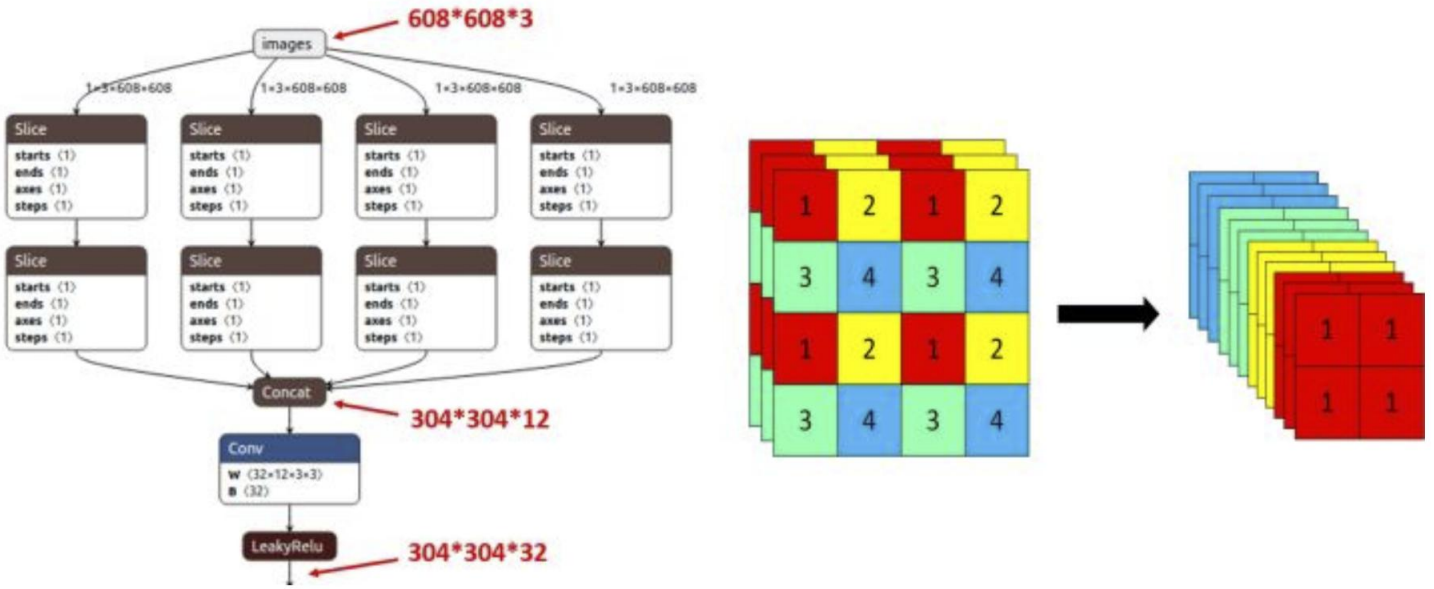
实验五：Yolo


608*608*3
Focus
CBL
CSP1_1
CSP1_3
SPP
CSP2_1
上采样
Concat
CONV
76*76*255
38*38*255
YOLOV5s Struture
CBL = Conv BN Leaky relu
Res unit = CBL CBL add
CSP1_X = CBL Res unit Conv Concat BN Leaky relu CBL
X个残差组件
CSP2_X = CBL CBL Conv Concat BN Leaky relu CBL
2*X个
Focus = slice slice slice slice Concat CBL

实验五：Yolo

Focus是Yolo v5新增的操作，右下图就是将4 * 4 * 3的图像切片后变成2 * 2 * 12的特征图。以Yolo v5s的结构为例，原始608 * 608 * 3的图像输入Focus结构，采用切片操作，先变成304 * 304 * 12的特征图，再经过一次32个卷积核的卷积操作，最终变成304 * 304 * 32的特征图。

需要注意的是：Yolo v5s的Focus结构最后使用了32个卷积核，而其他三种结构，使用的数量有所增加。



实验五：Yolo

Yolo v5的neck和Yolo v4中一样，都采用FPN+PAN的结构。

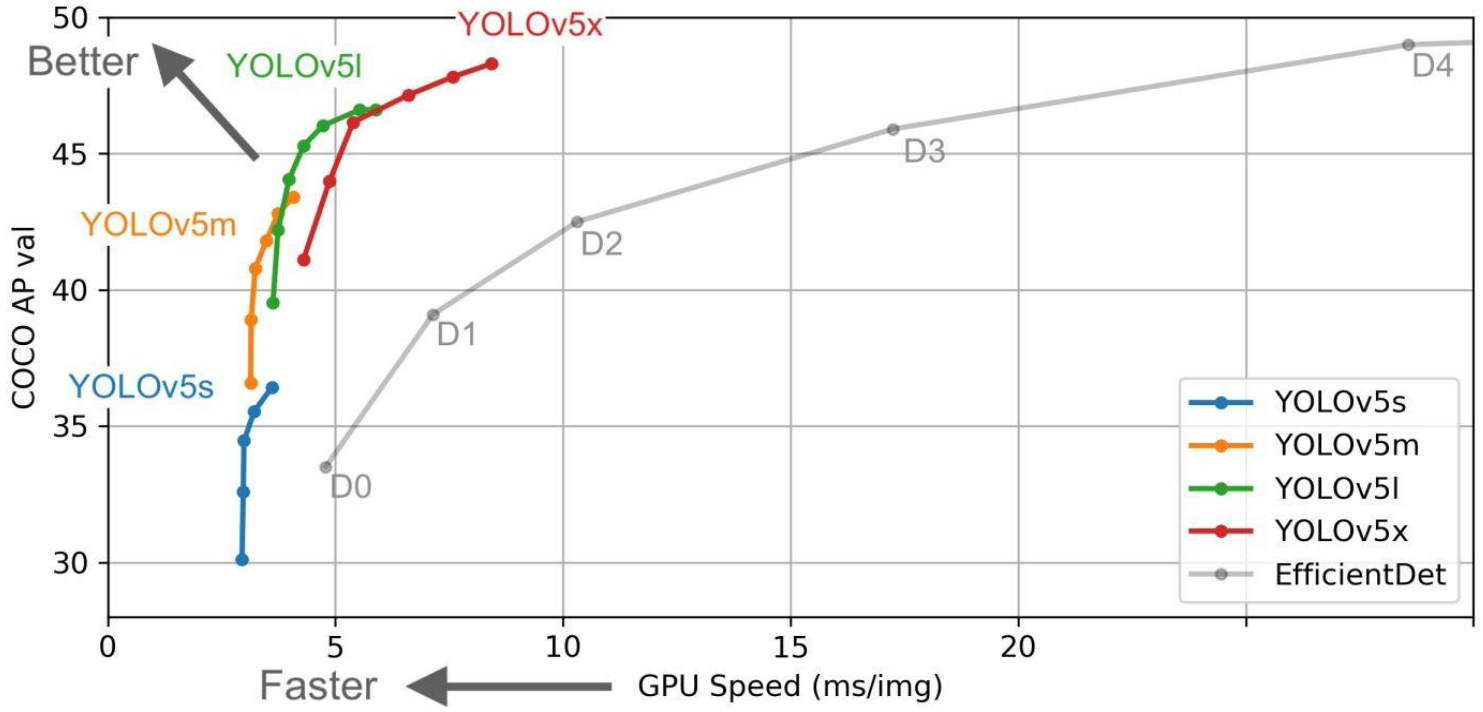
FPN是自顶向下，将高层的强语义特征传递下来，对整个金字塔进行增强，不过只增强了语义信息，对定位信息没有传递。PAN就是针对这一点，在FPN的后面添加一个自底向上的金字塔，对FPN补充，将低层的强定位特征传递上去。

Yolo v4的Neck结构中，采用的都是普通的卷积操作。而Yolo v5的Neck结构中，采用借鉴CSPnet设计的CSP2结构，加强网络特征融合的能力。

实验五：Yolo

Yolo v5s网络最小，速度最少检测的以大目标为主，追求速度。其他的三种网络，在此基础上，不断加深加宽网络，精度也不断提升，但速度的消耗也在不断增加。

目前，Yolo v5s的模型十几M大小，速度很快，线上生产效果可观，嵌入式设备可以使用。



参考：

1. You Only Look Once: Unified, Real-Time Object Detection

https://arxiv.org/abs/1506.02640

2. YOLO: Real-Time Object Detection

https://pjreddie.com/darknet/yolo/

3. YOLOv4: Optimal Speed and Accuracy of Object Detection

https://arxiv.org/abs/2004.10934

实验要求：

1. 要求自己编程实现YOLO（v3-v11任一种）检测算法；

2. 开发语言为python或者C/C++；

3. 要求提交报告和代码。