1. 卷积神经网络剪枝到底是怎么做的，是变成0还是cut掉。

这一块我们还在做，目前的做法是将元素置零处理，但是不太稳定。通过读论文发现还有其他剪枝的方法，比如每一轮迭代会将全部的卷积窗口进行排序（排序指标为卷积核中L1正则化的权重参数），舍弃排序后指标最低的m个卷积窗口以达到剪枝的目的，然后用剪枝后的卷积窗口进行模型训练，再不断地重复这个过程。

二值化：分解之前的每个权值矩阵为一个二值权值矩阵 与一个尺度系数（标量）（一个实数）的相乘，即。在训练这个二值网络之前，先训练一个标准的卷积神经网络，训练后得到根据指定任务（如目标检测、物体识别等）优化后的网络权重，其中，且这些权重为任意实数。再根据式（3）中的目标函数，来寻找可以使 最小的，意即让尽可能的表征优化好的权值矩阵： (3)。针对每一层每个卷积核的权值矩阵根据式（3）优化后得到相应的，进而可以得到整个网络的二值系数矩阵，以及二值权值矩阵的集合。根据已得到的 与 来端对端的训练网络，此时训练网络时更新的主要为系数矩阵 中的值。

1. 信号处理机是gpu还是cpu

NVIDIA在2014年推出了搭载GPU的嵌入式平台。Arm是计算力太差。Dsp计算能力强但是移植繁琐。

1. 随机森林和dropout的关系

都属于集成方法，dropout属于另类bagging。参加西瓜书p178.都具有随机性。都可以防止过拟合。

1. Adaboost

Adaboost是一种迭代算法，其核心思想是针对同一个训练集训练不同的分类器(弱分类器)，然后把这些弱分类器集合起来，构成一个更强的最终分类器(强分类器)。其算法本身是通过改变数据分布来实现的，它根据每次训练集之中每个样本的分类是否正确，以及上次的总体分类的准确率，来确定每个样本的权值。将修改过权值的新数据集送给下层分类器进行训练，最后将每次训练得到的分类器最后融合起来，作为最后的决策分类器。

更多参见西瓜书p173.

1. 多态，c++三大特性，虚函数

虚函数：<https://www.cnblogs.com/malecrab/p/5572730.html>

<https://www.cnblogs.com/jianyungsun/p/6361670.html>

<https://www.cnblogs.com/jin521/p/5602190.html>

1. 平衡二叉树，完全二叉树  
   平衡二叉树：在AVL中任何节点的两个儿子子树的高度最大差别为一

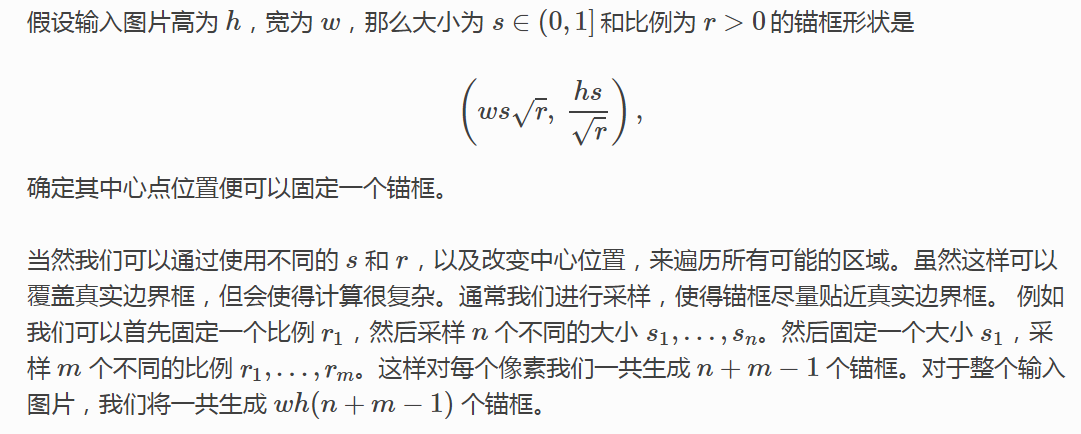
完全二叉树：若设二叉树的深度为h，除第 h 层外，其它各层 (1～h-1) 的结点数都达到最大个数，第 h 层所有的结点都连续集中在最左边。

完满二叉树：所有非叶子结点的度都是2。（只要你有孩子，你就必然是有两个孩子。）

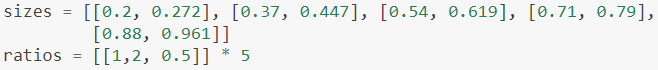
1. 面向对象编程和面向过程编程有啥区别

参见： <https://blog.csdn.net/bieleyang/article/details/78330362>

1. Ssd锚框的面积



Gluon中ssd锚框面积：



1. Kmeans如何防止陷入局部最优，模拟退火有了解吗

但是可以重复执行几次kmeans，选取SSE最小的一次作为最终的聚类结果。

模拟退火：<https://www.cnblogs.com/LBSer/p/4605904.html>

<https://www.cnblogs.com/hdu-2010/p/4322841.html>

<https://blog.csdn.net/qq_34374664/article/details/78332983>

<https://www.cnblogs.com/rvalue/p/8678318.html>

1. Relu的优点

第一，采用sigmoid等函数，算激活函数时（指数运算），计算量大，反向传播求误差梯度时，求导涉及除法，计算量相对大，而采用Relu激活函数，整个过程的计算量节省很多。

第二，对于深层网络，sigmoid函数反向传播时，很容易就会出现梯度消失的情况（在sigmoid接近饱和区时，变换太缓慢，导数趋于0，这种情况会造成信息丢失），从而无法完成深层网络的训练。

第三，Relu会使一部分神经元的输出为0，这样就造成了网络的稀疏性，并且减少了参数的相互依存关系，缓解了过拟合问题的发生

缺点：由于负数部分恒为0，会导致一些神经元无法激活（可通过设置小学习率部分解决）；输出不是以0为中心的

1. 激活函数有哪些

Sigmoid，tanh，relu，Leaky-ReLU，P-ReLU，R-ReLU，Maxout

参见：https://blog.csdn.net/u013146742/article/details/51986575

1. Dbscan的算法大概流程

参见总结

1. Dbscan和kmeans的优缺点

参见总结

1. Dssd中的反卷积层如何和巻积层结合在一起

高层特征图与低层特征图在通道融合的时候，使用了点乘，DSSD作者也使用过点加，结果发现通道之间相乘比相加可以提升0.2%个map，但是推理速度像素相加要略快于相乘。

1. 类别不均衡问题

参见：<https://blog.csdn.net/u013829973/article/details/77675147?locationNum=4&fps=1>

<https://blog.csdn.net/u013709270/article/details/72967462>

1. Dssd中的预测模块是怎么样的，为什么好

参见总结

1. DSSD做了什么改进，为什么要这样做

参见总结

1. Sigmoid函数形式

参见公式

1. 更新线性回归的方式(除了梯度下降)

最小二乘法

1. 指数平滑的作用

一阶指数平滑实际就是对历史数据的加权平均，适用于直线型数据；二次指数平滑是对一次指数平滑的再平滑。它适用于具有线性趋势的时间数列；三次指数平滑适用于具有季节性效应的数据

1. 对YOLO有了解吗，yolo的损失函数