pytorch实现VGG16的网络:
请参考:
https://blog.csdn.net/qq_40360172/article/details/109176612

faster RCNN中RPN相比之前做了什么优化:
请参考:
https://blog.csdn.net/weixin_30566111/article/details/96658575

dropout 是否用在测试集上
请参考:
https://zhuanlan.zhihu.com/p/118390256

YOLO v3进行了几次下采样
请参考:
https://blog.csdn.net/qq_31511955/article/details/87917308

列举几个梯度下降的方法
请参考:
https://blog.csdn.net/u012328159/article/details/80252012

Bert 类模型中的绝对位置 embedding 和 相对位置 embedding 怎么理解，各自的优缺点和使用场景
请参考:
https://blog.csdn.net/weixin_44799217/article/details/115374101

Bert 的预训练任务有哪些，各自的作用是什么
请参考:
https://www.cnblogs.com/wwj99/p/12283799.html

人体姿态估计主流的两个做法是啥？简单介绍下
请参考:
https://blog.csdn.net/zhou4411781/article/details/100370661

普通的逻辑回归能否用于大规模的广告点击率预估？为什么？

解析：

不能

第一，数据量太大。传统的逻辑回归参数训练过程都依靠牛顿法（Newton's Method）或者 L-BFGS 等算法。这些算法并不太容易在大规模数据上得以处理。

第二，不太容易得到比较稀疏（Sparse）的答案（Solution）。也就是说，虽然数据中特征的总数很多，但是对于单个数据点来说，有效特征是有限而且稀疏的。

9.卷积的实现原理以及如何快速高效实现局部weight sharing的卷积操作方式

- 卷积操作在具体实现时有多种方式
- 可以直接按照公式实现，但是这种方法无法并行计算。
- 利用高效得并行计算得方式来实现卷积

通常需要牺牲空间。

具体做法：

- 将卷积得第一个滑动窗内得数据取出，做为列向量v1。 ü 将卷积得第二个滑动窗内得数据取出，做为列向量v2。
- N 将卷积得第N个滑动窗内得数据取出，做为列向量v_N。
- 将v1,v2,v_N组合成NxK的矩阵与KxK得卷积核做矩阵乘法，即实现局部快速 weight sharing卷积。

注意到：v1,v2之间数据可能重复。这就多占用了空间。