本次实验我主要负责的是和刘志杰同学一起进行网络的搭建和训练。在网络搭建途中，我们使用了VGG和Resnet。我接下俩会着重写在选择优化算法时的考虑。

在优化参数的时候，我们有多个选择，因为pytorch会自带一些优化算法，因此我们使用了最常见，也是最简单的SGD。随机梯度优化算法相较于我们课堂上讲的梯度优化算法收敛速度更快，而且优化的方向也大致是向最优的方向优化的。

我们原本的梯度优化算法（批量梯度下降，简称BGD）本来是每更新一次参数都需要全部的训练数据。但是这种算法在面对大数据样本的时候，会显得非常无力，优化速度非常慢。这个时候，就需要随机优化算法。当数据量非常大的时候，一次考虑全部样本已经显得非常不明智。这个时候SGD提出了一种解决方法，就是对每个训练样本数据都作为一个训练数据集，来更新参数，对每一个训练数据都进行这样的操作，最后就可以得到结果。这个相对于之前的又换算法一下子考虑全局数据来说，虽然也看重全局的数据分布，但是已经弱化也一些。但是，我们也可以发现一个显著的问题，就是，本来的BGD是完全考虑全局数据的，优化的方向基本上就是往最优化或者局部最优的方向优化的。但是，如果对每一个数据都更新一次参数的话，整体的方向性虽然也会向最优的方向走，收到的噪声却比之前大，这些噪声就会影响到参数优化的方向，只能保证是大致向最优的方向优化的。由于在每次更新参数前都会对相似的样例进行梯度重复计算，批量度下降会在大数据集上做出很多的冗余计算。SGD每次计算一个样例的方式就能避开这种冗余。我们做出损失函数的波形的时候，会发现损失函数的值会发生剧烈波动。这是由于SGD频繁执行更新所伴随高方差导致的。在下降到最小值之前，SGD的剧烈波动会很难收敛到确切的最小值。SGD最重要的一个部分就是学习率的确定，SGD的收敛是和学习率挂钩的，当减小学习率时，SGD会几乎表现出和批量梯度下降相同的收敛行为。因此，学习率是整个项目的重中之重。



实际上，在考虑优化算法的时候，我们本来还考虑了遗传算法。遗传算法在全局的搜索能力还是比较好的，与之相伴的就是训练的时间会非常长。而且，如果使用遗传算法的话，我们的预选方案有两种，一种是随机生成初始种群，一种是根据我们给出的种子值结合3西格玛定律来生成种群并搜索最优解。在cma包中都有介绍，而且在这个包中还有自适应的遗传算法，相较于基础的遗传算法，这个收敛速度也更加快，加上调用包也是很简单，我们倾向于这一次新奇的尝试。但是问题马上来了，如果我们选择方案一，这就会使优化时间很长。如果使用方案二，快可能是快了，但是前提是我们能给出一组很好的种子值。而由于我们没有很好的参考条件，只能随机生成种子值。这就又回到了第一种选择，甚至因为种子值的设置，加上很大概率在3西格玛的领域内搜索，会导致陷入局部最优。这时候，如果我们将西格玛的值调大，搜索的范围确实会变大，但又会导致训练的时间加长。后来，我们在网上查找资料发现，有人尝试过使用这个算法对CNN进行优化过。但是优化的结果无一例外，基本上都是花费的时间很长，而且最后的准确率也不能保证，甚至是非常差的。鉴于此，我们放弃了使用遗传算法。

