# 训练过程中学习率变化思路

* 1. 传统方法
  2. 设定学习率：

图示

描述已自动生成

不同学习率对训练loss的影响。过高过低都不行

方法：

Fastai包：

learn.lr\_find()

learn.sched.plot\_lr()

对每次迭代的学习进行记录，并绘制学习率（对数尺度）与损失，我们会看到，随着学习率的提高，从某个点开始损失会停止下降并开始提高。

图表, 折线图

描述已自动生成 图表

描述已自动生成

Flowers db,vgg16 LR MNIST SimpleNN LR

理想的学习率，应该在开始下降到最低点之前的范围内

* 1. 问题

一般而言，当已经设定好学习速率并训练模型时，只有等学习速率随着时间的推移而下降，模型才能最终收敛。然而，随着梯度达到高原，训练损失会更难得到改善。

图表, 表面图

描述已自动生成

误差曲面中的鞍点。鞍点是函数上的导数为零但不是轴上局部极值的点

相比于局部最低点，鞍点更加阻碍收敛。如果鞍点正好发生在一个巧妙的平衡点，小的学习率通常不能产生足够大的梯度变化使其跳过该点（即使跳过，也需要花费很长时间）。这正是周期性高学习率的作用所在，它能够更快地跳过鞍点。

* 1. CLR周期循环改变学习率

图表, 折线图

描述已自动生成图表, 折线图

描述已自动生成

Triangular2 exp\_range

还有一种不递减的方法triangular

**设置：stepsize等于2~10的epoch的step，boundary（范围设为acc的顶点对应的lr为上限，下限为其1/3 ,1/4）**

结果：

表格

描述已自动生成

表格

描述已自动生成

表格

描述已自动生成

python models/KNet/net.py -b 64 --workers 4 --weights /data/resnet34-pretrain.pth --result /data/ngx/handtracking/result --epochs 15

# checkpoint train

python models/KNet/net.py -b 256 --CLR triangular2 --workers 8 --adam --result /my\_pros/ngx/handtracking/result --resume 79 --epochs 90

Knet 学习率设定：

前方法：lr warm\_up 然后每个epoch递减。CLR：step\_size=2xn\_iter(4个epoch一个循环)，范围（1e-3 – 6e-3）, 模式Triangular2

图表, 折线图

描述已自动生成下图平滑的是原方法，有明显周期性起伏的是CLR

图表, 折线图

描述已自动生成

图表, 折线图

描述已自动生成原方法略好一些，acc差0.1-0.2%

图表, 折线图

描述已自动生成

使用CLR exp\_range模式下结果轻微好于原方法，acc差别<0.1%。最终acc 96.76%

图形用户界面, 图表, 折线图

描述已自动生成

尝试使用Adam优化器+CLR（exp\_range）对比使用Adam，acc 96.9原方法轻微好于CLR,<0.1%，增加epoch同样结论。

结论：使用CLR方法改变学习率，可以作为一种备选思路，尝试结果，它能否使得模型收敛更快，loss更低没有一个确定的结论。