PyTorch

动量与学习率衰减

主讲人：龙良曲

Tricks

- momentum

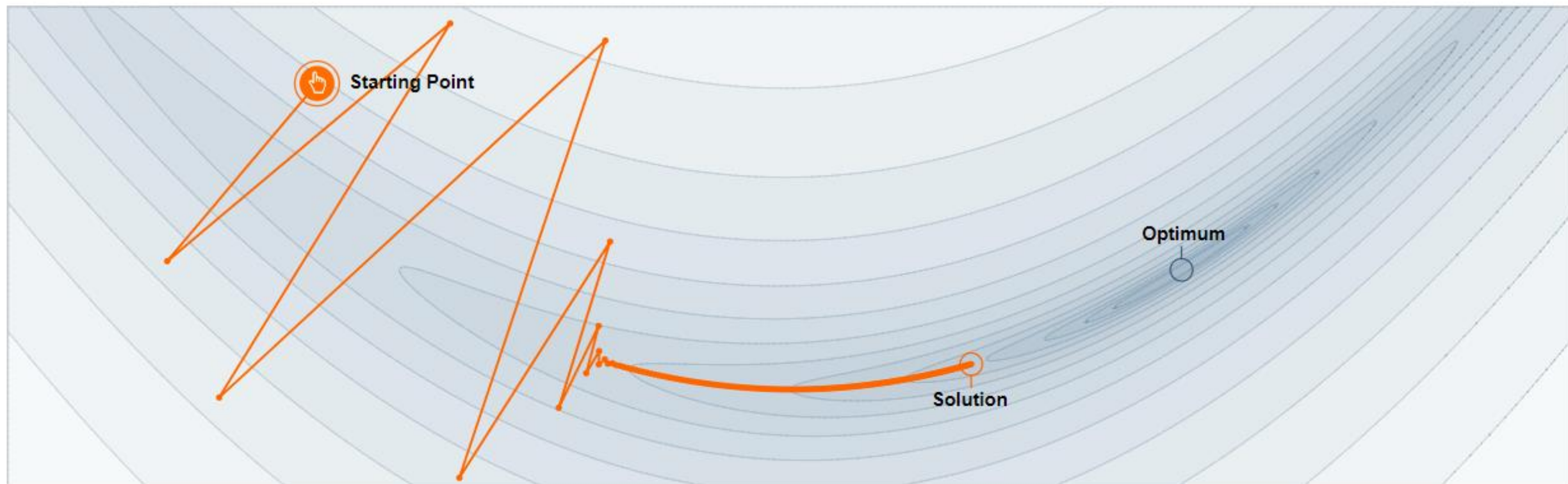- learning rate decay

Momentum

$$w^{k+1} = w^k - \alpha \nabla f(w^k).$$

$$z^{k+1} = \beta z^k + \nabla f(w^k)$$
$$w^{k+1} = w^k - \alpha z^{k+1}$$

No momentum

Step-size α = 0.0038

0   0.003   0.006

Momentum β = 0.0

0.00   0.500   0.990
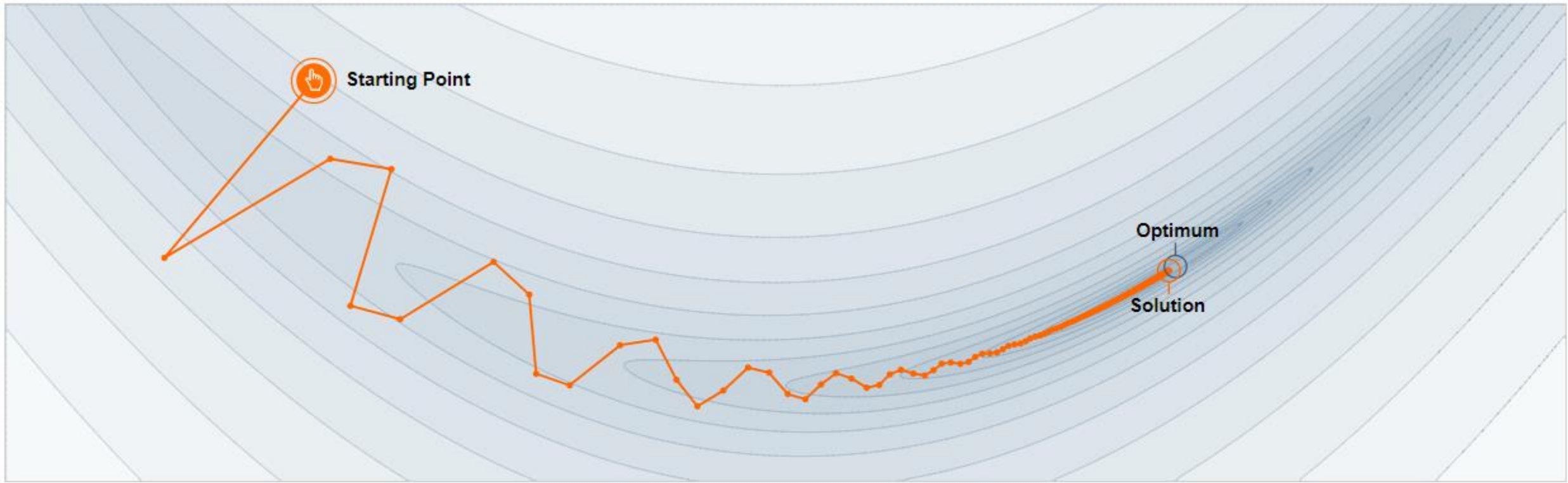
We often think of Momentum as a means of dampening oscillations and speeding up the iterations, leading to faster convergence. But it has other interesting behavior. It allows a larger range of step-sizes to be used, and creates its own oscillations. What is going on?

With appr. momentum

Step-size α = 0.0038

0   0.003   0.006

Momentum β = 0.78

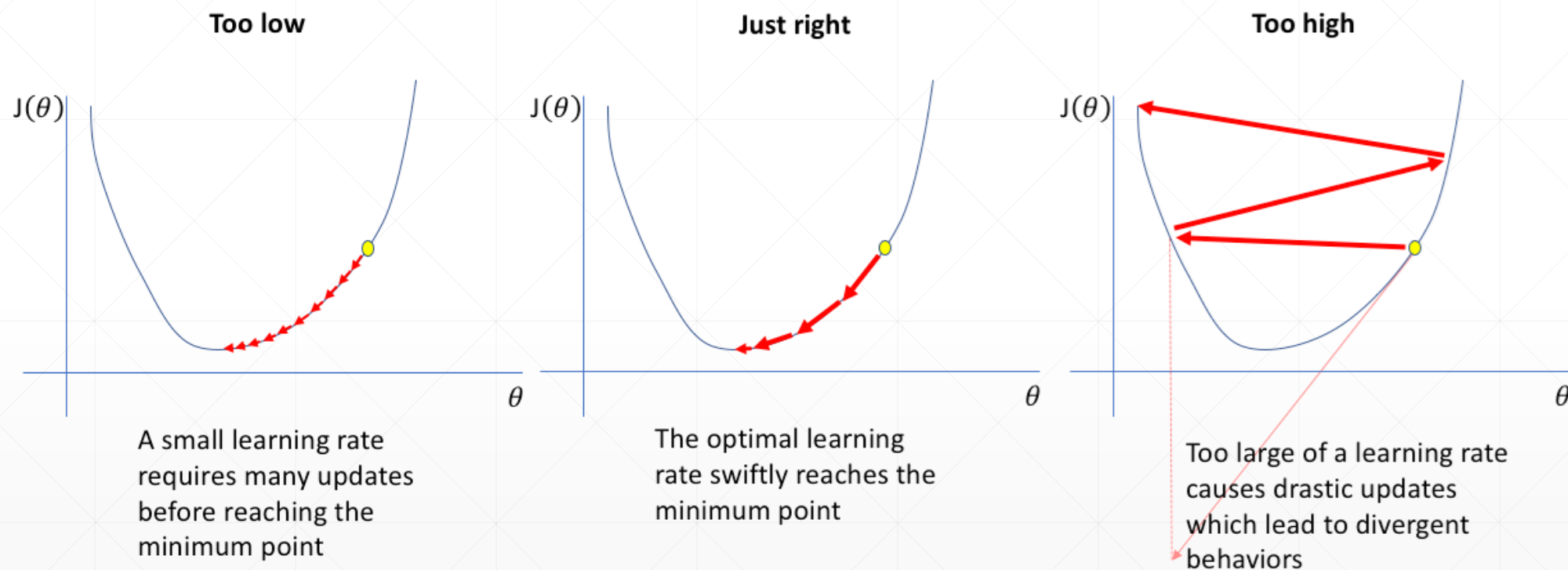0.00   0.500   0.990

We often think of Momentum as a means of dampening oscillations and speeding up the iterations, leading to faster convergence. But it has other interesting behavior. It allows a larger range of step-sizes to be used, and creates its own oscillations. What is going on?

momentum

```python
optimizer = torch.optim.SGD(model.parameters(), args.lr,
                                  momentum=args.momentum,
                                  weight_decay=args.weight_decay)
scheduler = ReduceLROnPlateau(optimizer, 'min')

for epoch in xrange(args.start_epoch, args.epochs):
    train(train_loader, model, criterion, optimizer, epoch)
    result_avg, loss_val = validate(val_loader, model, criterion, epoch)
    scheduler.step(loss_val)
```

Learning rate tunning

Too low

$J(\theta)$

θ

A small learning rate requires many updates before reaching the minimum point

Just right

$J(\theta)$

θ

The optimal learning rate swiftly reaches the minimum point

Too high

$J(\theta)$

θ

Too large of a learning rate causes drastic updates which lead to divergent behaviors

Andrej Karpathy ✓
@karpathy

3e-4 is the best learning rate for Adam, hands down.

♡ 408   11:01 AM - Nov 24, 2016   ⓘ

💬 124 people are talking about this   ⟩

Learning rate decay

Just right

$J(\theta)$

θ θ

The optimal learning
rate swiftly reaches the
minimum point

Step decay of learning rate

Learning rate

10^{-1}

10^{-2}

10^{-3}

Epoch

0   25   50   75   100   125   150   175   200
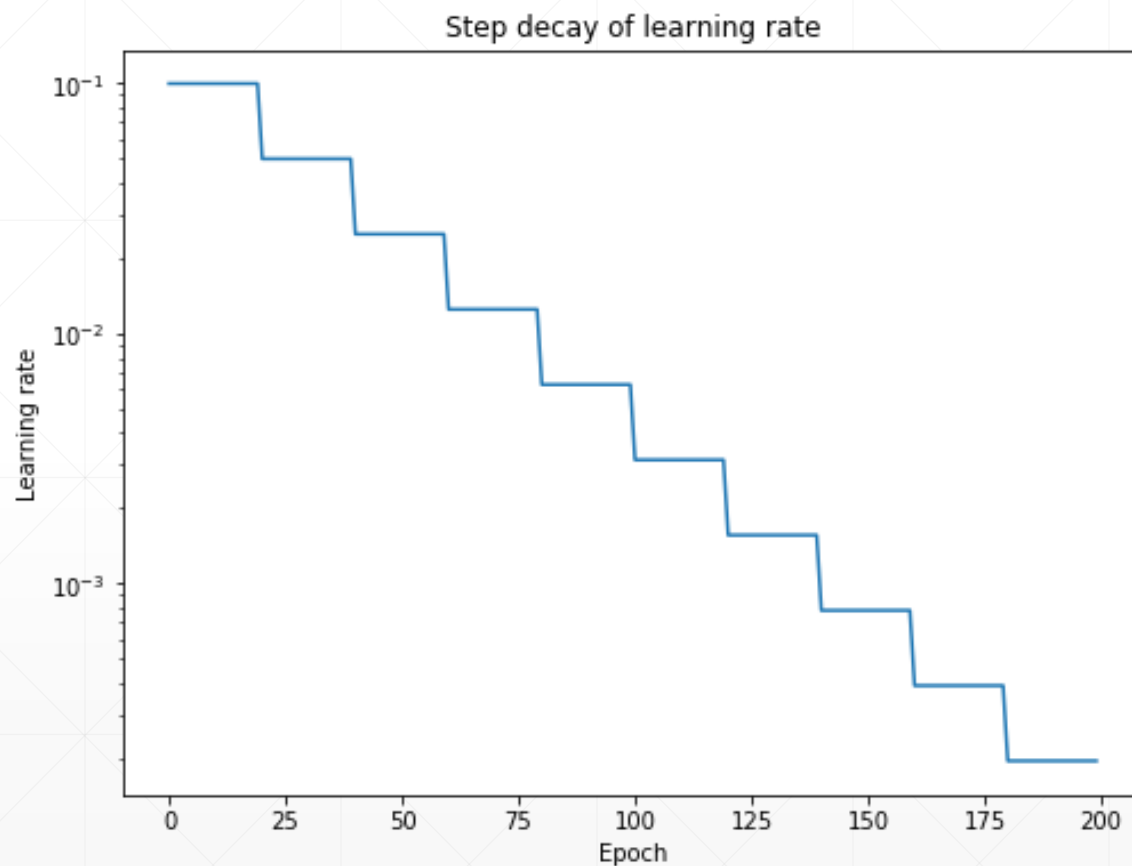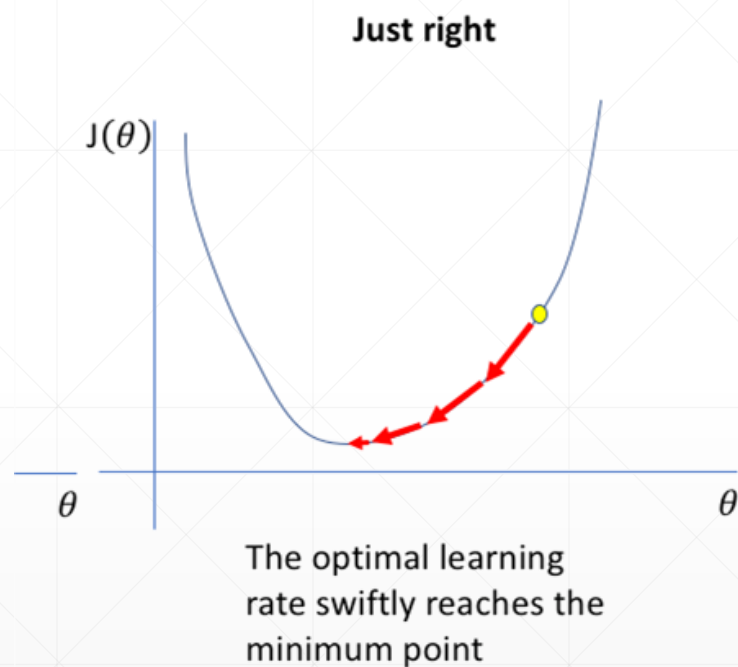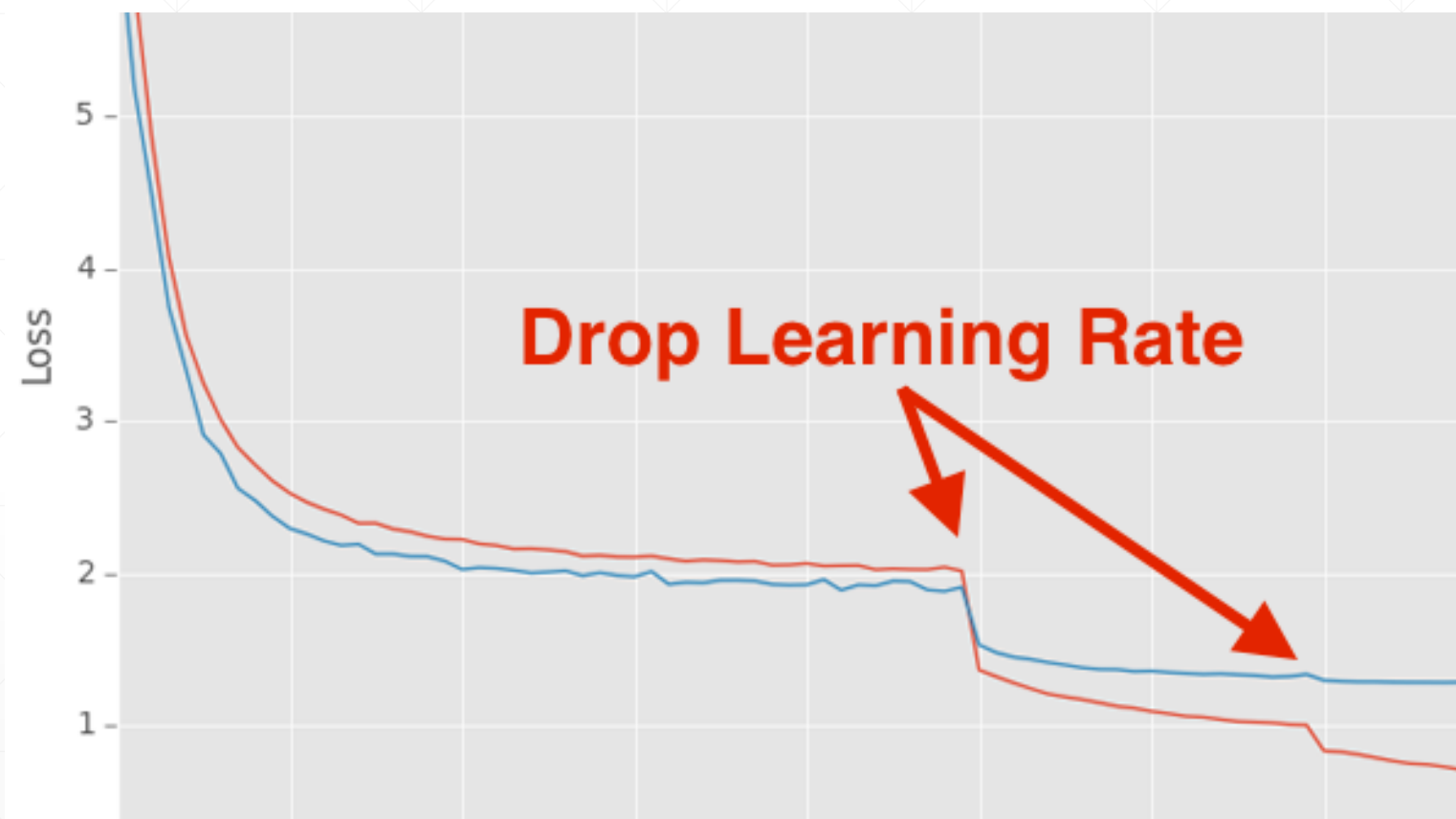
Scheme 1.

CLASS `torch.optim.lr_scheduler.ReduceLROnPlateau`(*optimizer, mode='min', factor=0.1, patience=10, verbose=False, threshold=0.0001, threshold_mode='rel', cooldown=0, min_lr=0, eps=1e-08*)

[SOURCE]

```python
optimizer = torch.optim.SGD(model.parameters(), args.lr,
                            momentum=args.momentum,
                            weight_decay=args.weight_decay)
scheduler = ReduceLROnPlateau(optimizer, 'min')

for epoch in xrange(args.start_epoch, args.epochs):
    train(train_loader, model, criterion, optimizer, epoch)
    result_avg, loss_val = validate(val_loader, model, criterion, epoch)
    scheduler.step(loss_val)
```

Scheme 2.

```python
# Assuming optimizer uses lr = 0.05 for all groups
# lr = 0.05      if epoch < 30
# lr = 0.005     if 30 <= epoch < 60
# lr = 0.0005    if 60 <= epoch < 90
# ...
scheduler = StepLR(optimizer, step_size=30, gamma=0.1)
for epoch in range(100):
    scheduler.step()
    train(...)
    validate(...)
```

下一课时

其他Tricks

Thank You.