国防科学技术大学
National University of Defense Technology

模式识别课程论文

使用 Mixed Precision 复现 ResNet50 在 ImageNet 上的精度

姓名：	**董佩杰**
学号：	20023104
专业：	**计算机科学与技术**
学院：	**计算机学院**

2020 年 12 月 26 日

目录

一、摘要

摘要：本篇论文主要研究如何复现 Deep Residual Learning for Image Recognition 论文中提到的 ResNet50 的精度，并研究使用 Mixed Precision 等训练方法对模型训练的影响。为了复现 ResNet50 网络在 ImageNet 验证集上的准确率，参考了许多方法，比如数据增强、warmup、调度器等方法，可以达到 75.63%的 top1 结果。在加速训练方面，总结了发现模型训练瓶颈的方法、常用的数据加速手段、分布式训练策略等。

关键词：卷积神经网络、ImageNet、图像识别、分布式训练

二、任务说明

ImageNet 是斯坦福大学李飞飞教授于 2012 年推出的图像分类数据集，其中包含了超过 1500 万个标记过的高分辨率图像，具有 22,000 个类别。其中 ImageNet 大规模视觉识别挑战赛（ILSVRC）就是 Pascal 视觉挑战赛的重要组成部分，该比赛采取了 ImageNet 的一个子集，由 1000 个类别、120 万张训练图像、5 万张验证图像和 15 万张测试图像组成，约 150GB。

在 ImageNet ILSVRC 2012 中，评价指标通常是两个错误率 top1 和 top5。以 top5 错误率为例，它代表当给定一张图片，正确的标签不在模型预测的 5 个图片最可能的类别之中的分数。

本文将设计并实现一个经典的分类器 ResNet，在百万张可公开获取的图像数据集 ImageNet ILSVRC 2012 上完成分布式训练、推理等任务。

三、数据处理

数据处理在深度学习中有着极为重要的地位，数据集的大小、数据标注的质量、图像的分辨率、图像的格式、数据增强方法等对模型的性能有着至关重要的影响。这个部分参考 Bag of Tricks for Image Classification with Convolutional Neural Networks 论文和 Deep Residual Learning for Image Recognition 中的策略。

在训练过程的预处理中，采用了以下四个步骤对数据进行增强：

1) 随机采样一张图片，将其解码为 32 位浮点数，每个像素的值在 0 到 255 之间。

2) 随机调整图片大小，然后裁剪出 224x224 大小的图片。

3) 以 0.5 的概率随机水平翻转该图片。

4) 进行归一化，减去均值 [123.68,116.779,103.939]，然后除以方差 [58.393,57.12,57.375]。

在验证过程的预处理中，采用了以下三个步骤对数据进行处理：

1) 先将图片调整大小到 256x256。

2) 然后进行中心裁剪方法，裁剪出 224x224 大小的图片。

3) 对 RGB 通道进行均值化操作。

四、算法原理

本次实验选择了卷积神经网络发展历史中最重要的模型之一，ResNet 作为核心的模型。Deep Residual Learning for Image Recognition 这篇文章中提出的核心思想就是残差学习，通过残差学习可以让模型变得更深，从而解决深度网络退化问题。根据网络规模，将其从小到大提出了 ResNet18、ResNet34、ResNet50、ResNet101 和 ResNet152 共五个模型，如下图所示，模型之间采用的 block 以及 block 重复的次数略有不同。其中 ResNet50 是其中中等规模的模型，参数量、模型的容量适中，经常用作研究的基线模型，所以本实验也将使用 ResNet50 模型，并对其结果进行复现。

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

图 4-1 ResNet **系列模型结构（图源** https://arxiv.org/pdf/1512.03385v1）

VGGNet 网络通过加深网络取得了比浅层网络更好的性能，但是实验发现 VGG19 性能要比 VGG16 更差，越深的网络反而性能更差。ResNet 中也提出了一个 plain 网络来验证这个想法，如下图所示：

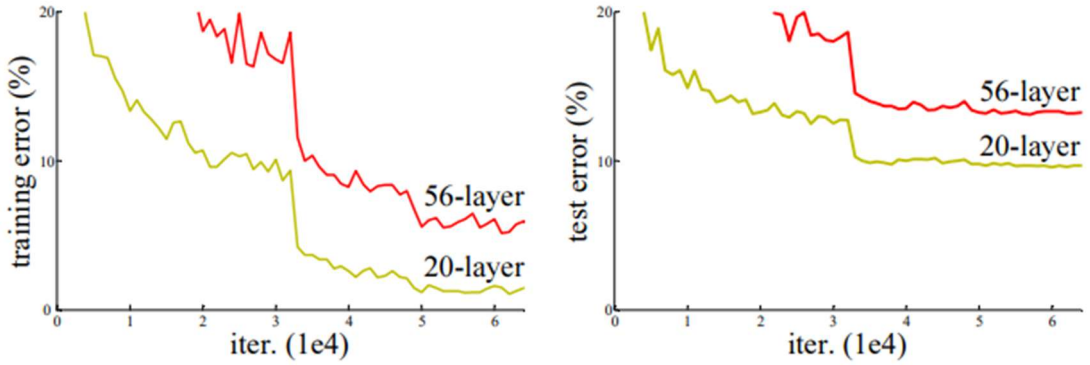

图 4-2 Plain 网络在 CIFAR-10 上的训练错误和测试错误曲线

从图 4-2 可以看出无论是训练误差还是测试误差，56 层的 plain 网络的表现都差于 20 层的网络，这就是网络退化的问题，即随着网络深度的增加，准确率开始饱和，然后快速下跌。在增加网络深度的情况下，保持准确率不下降的一个最简单的想法就是复制浅层的特征，即恒等映射。这也就是残差学习的来源，提出了残差学习单元，如下图 4-3 所示。

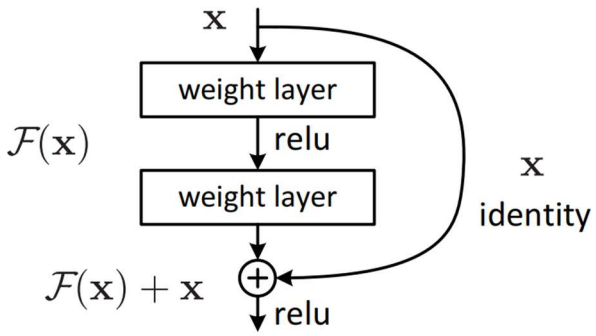

图 4-3 残差学习单元

上图中，x 代表输入特征，$F(x)$代表经过左侧路径以后学习到的内容。在加入了恒等映射以后，相当于目前整个单元需要学习的内容是$F(x)+x$，而 x 是不需要进行学习的，只需要学习$F(x)$，残差学习要比直接学习原始特征要容易，所以引入残差学习单元后，性能不会下降，因而可以设计更深的网络。

ResNet 中设计了两种单元，如图 4-4 所示。

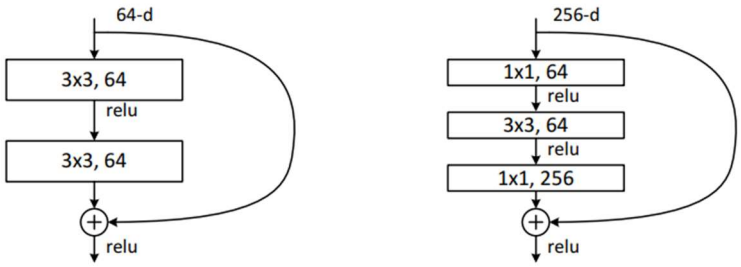

图 4-4 ResNet 中两种单元

ResNet18、ResNet34 使用的是左侧的残差单元，包含两个 3x3 卷积。ResNet50、ResNet101、ResNet152 使用的是右侧的残差单元，称为 bottleneck，使用了 1x1 卷积来降低模型的计算量。

五、具体实现

5.1 实验环境

硬件：

- CPU：Intel Core I9-10900K @ 3.70GHz
- 显卡：GeForce RTX 2080Ti(11G) x 4
- 内存：64G

软件：

- 系统：Window 10
- CUDA 11.0+CuDNN 8.0
- PyTorch 1.6.0、Apex、PyTorch Lightning
- 容器：Anaconda
- IDE：Vscode

5.2 实验设置

权重初始化设置为，对所有卷积层，使用 kaiming norm 方法进行初始化；对所有 BatchNorm 层，γ参数初始化为 1，β参数初始化为 0。

训练过程使用的优化器为 SGD，初始化 learning rate 为 0.1，其对应的 batch size 应该是 256，学习率和 batch size 的设置应该满足以上的比例关系。此外，momentum 参数被设置为 0.9，weight_decay 参数被设置为 1e-4。此外，在 epoch 为 30,60,90 的时候，学习率的值衰减十倍。

此外，还采用了 warmup 的策略，在前 5 个 epoch 使用 warmup 可以让训练更加稳定。具体做法采用的是渐进 warmup，让 learning rate 以线性的速度逼近设置的初始学习率。

$$warmup\ learning\ rate = \frac{global\ step}{5\ epoch\ step} \times base\ learning\ rate$$

5.3 训练加速

在训练大型数据集的时候，整个给过程可能存在一定的瓶颈，这时训练加速往往变得很

重要。 瓶颈可能在于 CPU、内存、GPU、IO 等地方，通常来说，主要有以下几个方法来进行数据加速。

图像解码，解码库非常丰富有 Pillow、Opencv、LMDB、Pillow-SIMD、TFRecords、TurboJpeg、jpeg4py 等，主要库的对比如下。PyTorch 中主要使用 Pillow 进行解码，解码速度上存在一定劣势，所以可以选择 Opencv 等库替换掉 Pillow，从而可以加快解码速度。

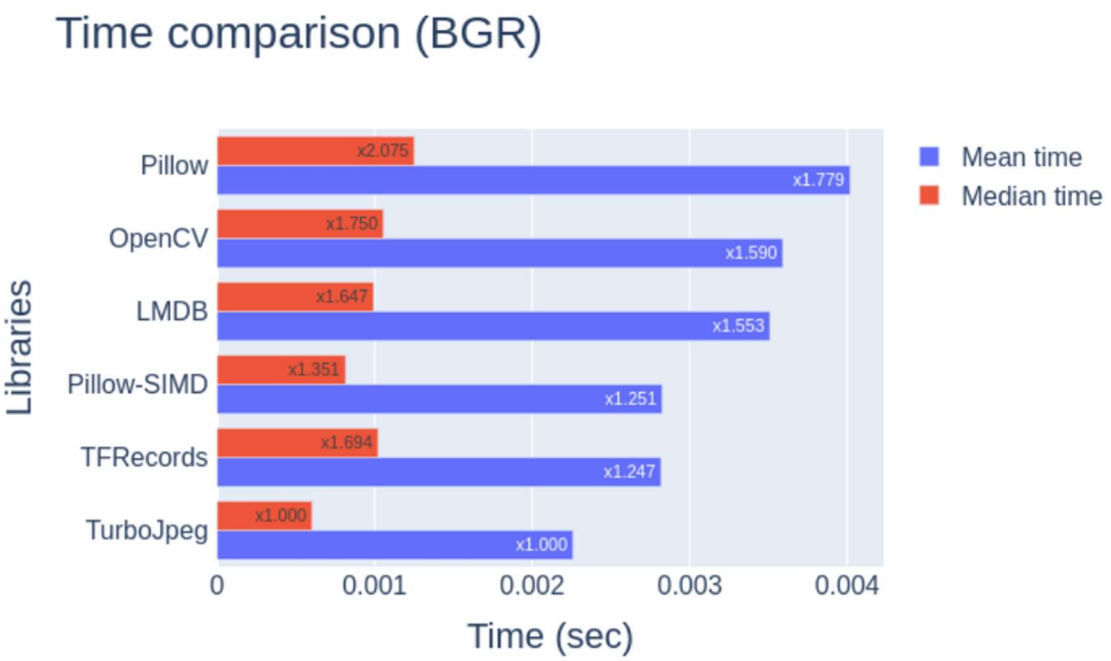

图 5-1 各个库解码时间对比

数据增强部分往往会对 CPU 造成比较大的压力，如果采用的数据增强方法过多，过于复杂，可以使用英伟达提供的 DALI 库进行开发，将数据加载、数据增强部分的工作在 GPU 端完成，可以极大提高训练效率。

数据预读取技术（data prefetch）是一种常用的加速手段，可以预读取下一次迭代所需要的数据，这样就不会出现 GPU 空等的现象，可以有效提高训练速度。

5.4 混合精度训练

混合精度是一项非常简单并且使用的技术，由百度和谷歌联合发表于 ICLR2018，可以让模型以半精度的方式训练模型，既能够降低显存占用，又可以保持精度。其核心就是使用了 IEEE 标准半精度(FP16)替代 FP32 表示的数据。使用半精度可以减少一半的比特，降低了算法带宽和内存带宽，能够提高处理器的吞吐量。经测试，使用了半精度以后吞吐量可以达到单精度的 2-8 倍，同时显存占用、内存占用几乎减半，可以用更大的 batch size 来训练模

型。

半精度也存在一定的问题需要解决，由于半精度可表示范围比较小，可能导致模型的精度损失。不过 mixed precision 论文中也提到了三种解决办法来规避精度损失，具体内容如图 5-2 所示。

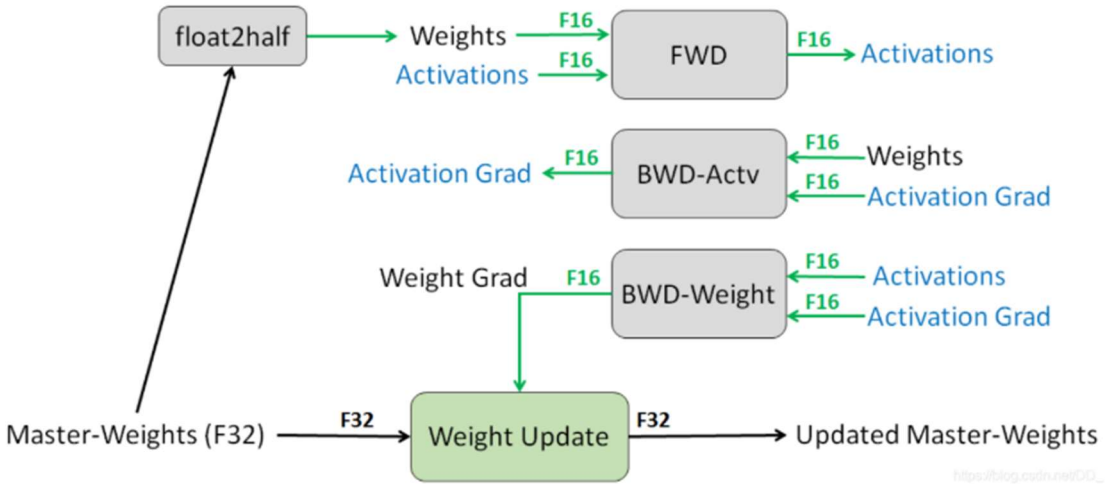

图 5-2 半精度训练流程

权重的更新需要使用 FP32 来进行更新，防止出现下溢出的问题，同时也可以尽量减少舍入误差带来的影响。Loss Scale 方法也是非常有效的方法，可以通过放缩 FP16 的值将网络实际的值平移到 FP16 可以表示的范围中，在 loss 更新的时候，除以放缩值，就可以达到 FP32 的精度。此外，向量点乘、Reduction、逐点操作都会先转成 FP32 的格式，然后转为 FP16 格式。通过以上三种方法，可以做到和 FP32 一样的效果，大幅度提高了模型的训练效率。

六、实验结果

ResNet50 参数量在 25.56M，每秒浮点运算数为 3.53G。具体训练过程中的结果如下图所示。

上图 6-1 到图 6-6 展示的内容是训练集、验证集上的 loss 曲线、top1 准确率曲线和 top5 准确率曲线。

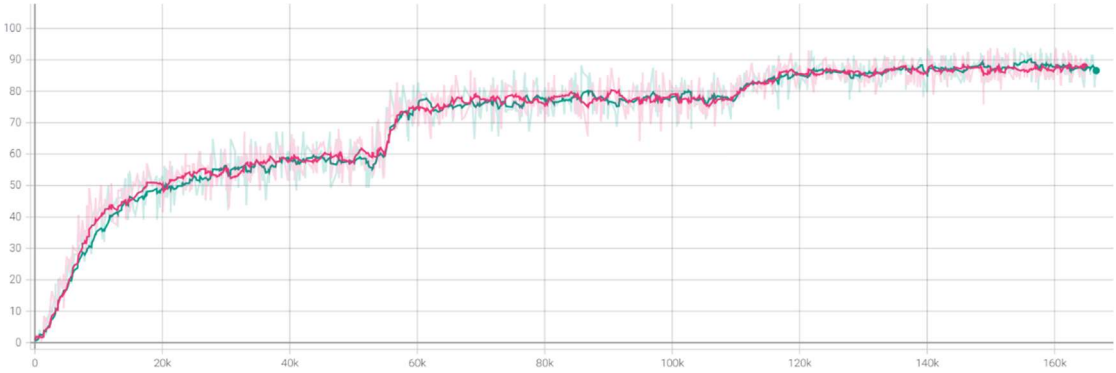

图 6-1 训练过程中 top1 准确率的实时结果展示

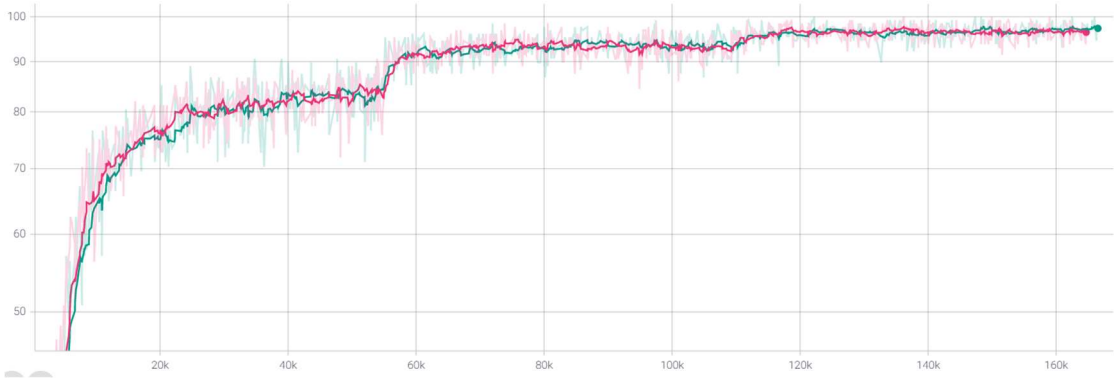

图 6-2 训练过程中 top5 准确率的实时结果展示

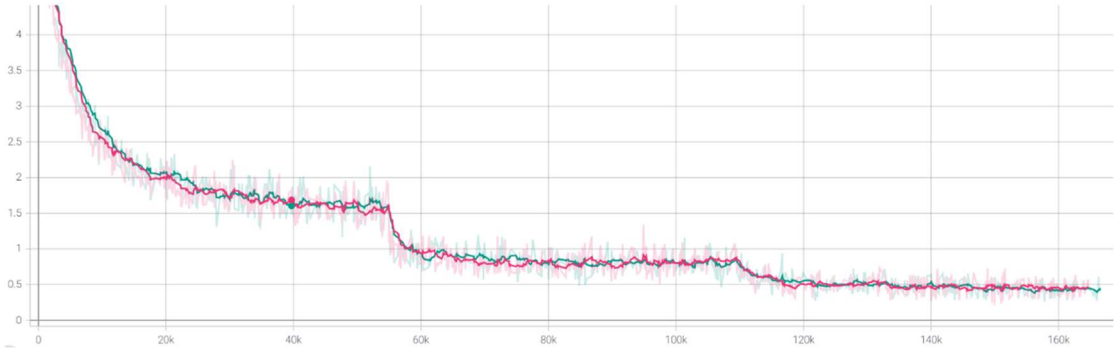

图 6-3 训练过程中实时 loss 曲线展示

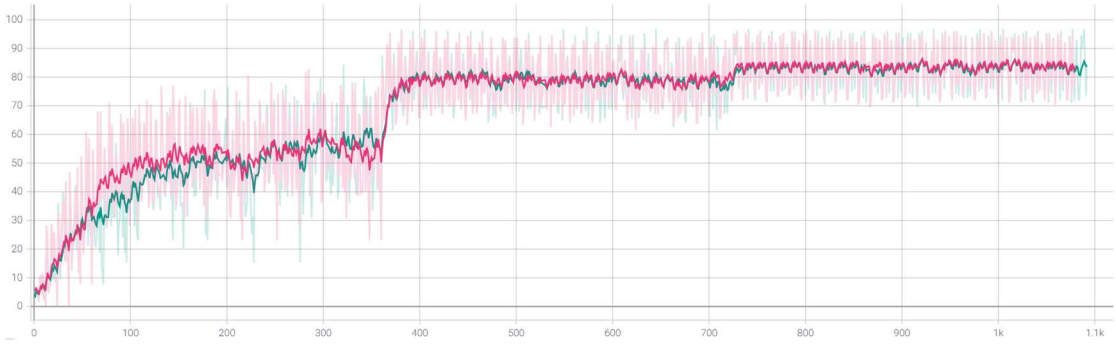

图 6-4 验证集上每个 batch 的 top1 准确率结果展示

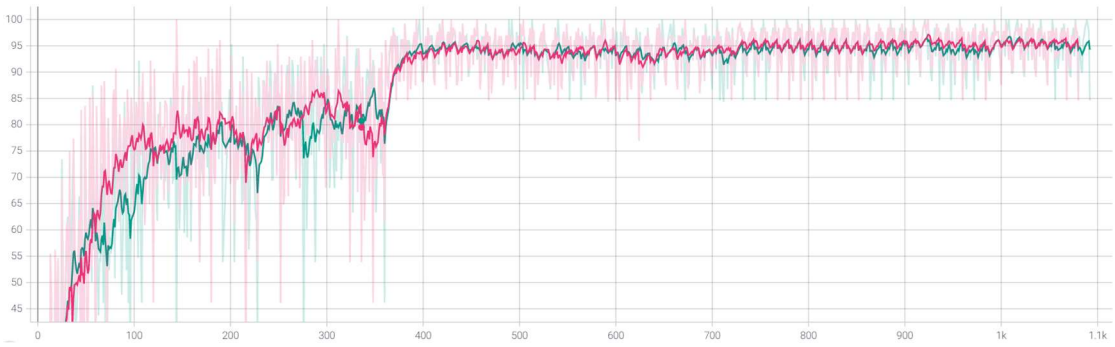

图 6-5 验证集上每个 batch 的 top5 准确率结果展示

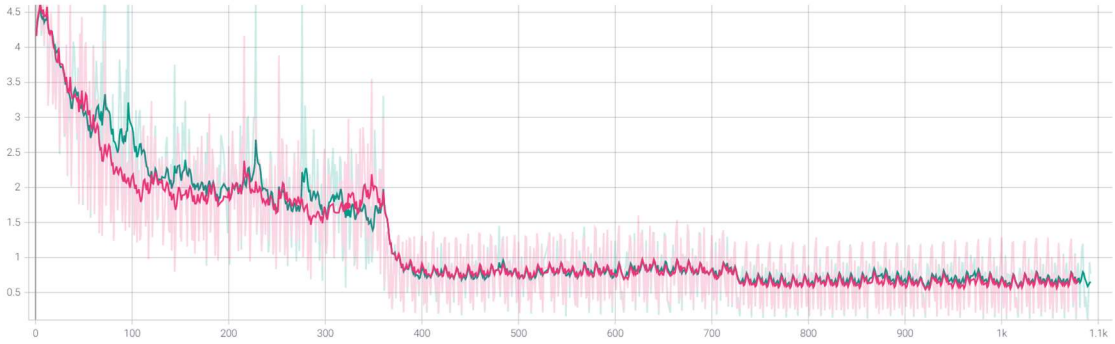

图 6-6 验证集上每个 batch 的 loss 曲线实时结果展示

从曲线中可以看出，大致上分为三个阶段，每个阶段之间都有一个跳变，这是 scheduler 调节学习率以后的效果，能让模型跳出局部最优点。

实验结果对比：

指标	混合精度	FP32 训练
Top1 Accuracy(train)	79.19%	78.94%
Top2 Accuracy(train)	92.78%	92.62%
Top1 Accuracy(val)	75.63%	75.46%
Top2 Accuracy(val)	92.68%	92.61%
Time-consuming	2day	2day12h

由上表结果可以看出，使用混合精度和 FP32 训练结果非常接近，差距在±0.3%以内，但是使用混合精度相比 FP32 训练可以节省 20%的时间。

经过本次实验可以得知，ResNet50 核心部件就是残差模块，通过残差模块可以让模型变得更深，避免网络退化的问题。同时本实验遵从了论文中的参数设置，在误差允许的范围内，做到了 ResNet50 在 ImageNet 验证集上的结果复现。在复现过程中遇到了有关效率的问题，由于 ImageNet 由 160G 之大，尽管用了 4 卡 2080Ti 进行训练，仍需要很长时间进行训练，所以对训练的优化就显得非常重要。本文探索了一系列 PyTorch 加速的方法，从数据读取、解码、多线程等角度加速了数据的读取，可以让显卡利用率保持在较高的水平，从而

实现了模型的加速。此外，本文核心就是测试 mixed precision 训练对结果的影响，通过结果证明，使用混合精度进行训练能节省 20%的时间，并且精度甚至略高于 FP32 训练的结果。

七、参考文献

[1] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing and Sun, Jian Deep Residual Learning for Image Recognition. (2015). , cite arxiv:1512.03385Comment: Tech report .

[2] Micikevicius, P.; Narang, S.; Alben, J.; Diamos, G.; Elsen, E.; Garcia, D.; Ginsburg, B.; Houston, M.; Kuchaiev, O.; Venkatesh, G. & Wu, H. (2017), 'Mixed Precision Training' , cite arxiv:1710.03740Comment: Published as a conference paper at ICLR 2018 .

[3] He, T.; Zhang, Z.; Zhang, H.; Zhang, Z.; Xie, J. & Li, M. (2018), 'Bag of Tricks for Image Classification with Convolutional Neural
Networks', .