- NLP方向总结-训练
  - 单机单卡GPU训练
    - 模型拷贝
      - model.cuda()
      - 模型是inplace的形式
    - 数据拷贝，每个step分别进行
      - data = data.cuda()
      - 数据是复制的形式
    - torch.cuda.is_available()判断GPU是否可用
    - 模型保存和加载
      - torch.save保存模型model.state_dict()，优化器，配置等
      - torch.load(file.pt, map_location=）。可以通过map_location直接加载到某个设备上，也可以用上面的模型拷贝方式加载后拷贝到设备上
  - 单机多卡
    - 检测GPU数量和控制使用GPU
      - torch.cuda.device_count()
      - 命令行环境变量: CUDA_VISIBLE_DEVICES控制使用哪些GPU
    - 串行数据的方式: torch.nn.DataParallel
      - 单进程多卡训练
      - 训练时包裹model即可
        - model = DataParallel(mode.cuda(), device_ids=[0,1,2,3])
      - 模型保存
        - torch.save时保存model.module.state_dict()
      - 模型加载
        - 没什么区别，torch.load即可
      - batch_size等于多个GPU的batch_size之和
    - 数据并行的分布式训练(推荐): torch.nn.parallel.DistributedDataParallel
      - 多进程多卡训练
      - 代码流程
        - torch.distributed.init_process_group(“nccl”, word_size=args.n_gpus, rank=args.local_rank)
        - torch.cuda.set_device(args.local_rank) 相当于CUDA_VISIBLE_DEVICES
        - 模型放到GPU上
          - model = DistributedDataParallel(model.cuda(args.local_rank), device_ids=[args.local_rank])
        - train_sampler =DistributedSampler(train_dataset)
        - train_dataloader = DataLoader(…,sampler=train_sampler)
        - 每个step的数据: data = data.cuda(args.local_rank)
      - 训练命令
        - python –m torch.distributed.launch —nproc_per_node=n_gpus train.py
      - 模型保存
        - torch.save在local_rank=0的位置保存，调用model.module.state_dict()保存模型参数
      - 注意
        - 每个进程的batch_size是这个GPU的batch_size
        - 每个epoch开始时，可以用train_sampler.set_epoch(epoch)充分打乱数据
        - 有了sampler, 就不用在DataLoader中设置shuffle=True了，互斥操作
  - 多机多卡
    - 数据并行的分布式训练(推荐): torch.nn.parallel.DistributedDataParallel
      - 代码和单机多卡一致
      - 训练命令不一样，用2台机器为例，每个机器有n_gpus个GPU，仅node_rank不一致，其它一样
        - python –m torch.distribute=ngpus —nnodes=2 —node_rank=0 —master_addr=“主节点IP” —master_port=主节点通信端口 train.py
        - python –m torch.distribute=ngpus —nnodes=2 —node_rank=1 —master_addr=“主节点IP” —master_port=主节点通信端口 train.py
      - 模型保存和加载，和单机多卡一致