Lab2 深度神经网络

北京航空航天大学计算机学院

1　实验目标

搭建深度神经网络，并在给定数据集上进行训练与测试。具体要求：

- 使用深度学习框架（TensorFlow、PyTorch、Caffe 等）完成网络搭建，推荐使用 tensorflow-1.15
- 不限制编程语言，推荐使用 Python 语言
- 数据集可以使用提供的 MINIST 数据集，也可自行调用神经网络框架相应接口下载并加载 MINIST 数据集
- 可使用提供的模板项目（TF-Mnist-Template），在其基础上进行修改完善，也可以重新编写代码。模板提供了框架，包括数据 IO 与模型训练以及测试，同学们的主要任务为模型定义与参数调整

2　数据集获取与使用

本次实验采用 Mnist 官方数据集[1]，模板项目的 mnist_data 目录下亦有相关文件。数据集和 lab-1 中数据集一致，此处不再赘述。

3　代码介绍

3.1　代码结构

模板项目基于 tensorflow-1.15.0 编写，代码结构如下：

- input_data.py：读取数据集文件，用于加载 mnist 数据集
- tf_minist.py：主文件，包含训练、测试流程以及模型保存的实现
- data_object.py：数据类文件，包含数据处理的实现
- tf_network.py：网络结构文件，包含网络结构实现
- tf_test_model.py：测试参数固化后保存的模型准确率

3.2　网络结构

LeNet 模型结构如下图 1所示，网络由 2 层卷积层、2 层池化层、2 层全连接层组成。可以发现，该网络输入与 Mnist 数据集不匹配，同学们可以采用两种解决方式：1）对输入图片周围补零（Padding 操作）至网络输入大小；2）修改第一层卷积层，使其输出与图中 C1 大小一致。更多网络细节可参考链接[2]。

LeNet 模型结构定义在 TF-Mnist-Template/tf_network.py 文件中，其中需要自行补充的函数为 Lenet 类中 net 函数。

net 函数被 forward 前传函数调用，包括了网络结构定义和前向传播两个过程，函数中包含 feats 参数，代表输入的图像数据，返回值则为前向传播后输出的 logits 结果。

[1] http://yann.lecun.com/exdb/mnist/

[2] https://www.cnblogs.com/wuliytTaotao/p/9544625.html

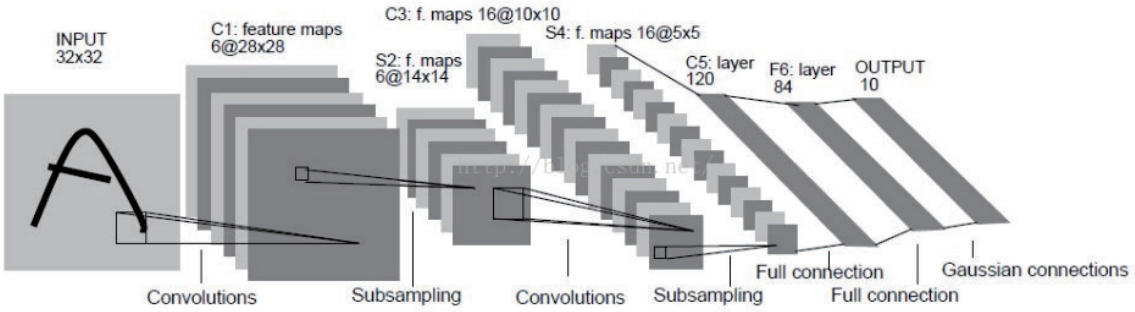

图 1: LeNet 模型结构图

```
def net(self, feats):
    """
    Define network.
    You can use init_weight() and init_bias() function to init
        weight matrix,
    for example:
        conv1_W = self.init_weight((3, 3, 1, 6))
        conv1_b = self.init_bias(6)
    :param feats: input features
    :return: logits
    """
```

3.3 代码流程

LeNet 模型训练与测试流程在 TF-Mnist-Template/tf_network.py 文件中，主要用于对 LeNet 模型进行初始化、模型训练以及测试。

在 main 函数中，程序建立了 session，调用 train_net 函数进行模型训练，test_net 函数进行模型测试。在 train_net 函数中，训练完成的模型保存路径为 model/model.pb

```
with tf.Session() as sess:
    ...
    train_net(data.train, data.validation)
    print("Training time: {:.3f}s.\n".format(time.time() -
        start_training_time))
    ...
    acc = test_net(data.test)
    print("Test Accuracy = {:.5f}".format(acc))
    print("Testing time: {:.5f}s\n".format(time.time() -
        start_testing_time))
```

而在 tf_test_model.py 文件中，使用者可以载入已保存的模型进行模型测试，评估模型的性能。

```
with tf.Session() as sess:
    with gfile.FastGFile('./model/model.pb', 'rb') as f:
        ...
    data = provide_data(mnist)
    ...
    # get accuracy and loss
    test_accuracy, loss = test_model(data.test)

    print("Test Loss = {:.5f}, Validation Accuracy = {:.5f}\n"
       .format(loss, test_accuracy))
```

4　作业要求

实现 tf_network.py 中的 LeNet 类，使用 tf_network.py 能够正常运行，网络要求结构如下所示：

- 网络输入：(batch_size, 28, 28, 1) 的数组，其中 batch_size 为每批次中包含图片的数量，这个数值可以根据自己硬件条件进行确定；28 * 28 为给定的图片尺寸。
- 网络输出：10 个输出节点，分别代表 0~9 这 10 个数字。本次作业不对精度做特别的要求，只需在合理范围内 (>80%) 即可。
- 网络模型：建议采用 LeNet，也可以采用其他自己定义的网络模型 (可适当调整网络以适配数据集输入维度大小)，调参并对比效果。

5　作业提交

- 若程序正常运行，则模型的结构与参数将会保存至 model/model.pb 中，无需进行额外操作
- 撰写实验报告，包括但不限于网络原理说明、网络实现代码介绍、训练过程截图、测试精度结果截图等。报告命名格式：学号 + 姓名 + 作业二报告.docx(.pdf)，如 XXXXXXXXX+张三 + 作业二报告.docx(.pdf)(更多细节参考“实验报告撰写格式”)
- 将完整的工程文件夹 TF-Mnist-Template (**需要包含模型 model 文件夹，不需要数据文件夹 mnist_data**)、实验报告打包成压缩文件。压缩文件命名格式：学号 + 姓名，如 XXXXXXXXX+张三 + 作业二.zip(.rar)。

6　其他

6.1　模型训练说明

本次作业使用的 MNIST 数据集中包含训练集 60000 张图片，如果在 CPU 上训练时间过长，可以考虑采用 GPU，或者减少训练集的大小（但会降低训练后的模型精度）。

6.2　Tensorboard 数据可视化

TensorBoard 提供机器学习实验所需的可视化功能和工具，建议学习和使用该工具[3]：

[3]https://www.tensorflow.org/tensorboard/get_started?hl=zh-cn

模板代码中已经写入了 tensorboard 相关语句，程序运行结束后工程目录下将会生成 logs 文件夹，接下来执行以下两步打开 tensorboard：

- 在终端中输入：tensorboard −−logdir=./logs ，如图 2所示
- 按照提示在浏览器中输入获得的网址打开 tensorboard

如图 3所示即是打开后的示例，能够清晰地观察训练过程 loss 以及 acc 变化曲线：

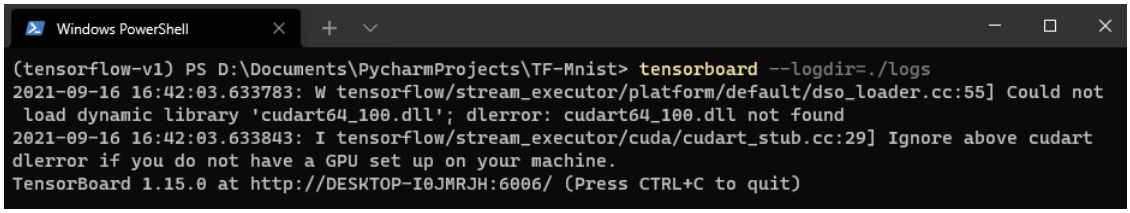

图 2: 打开 Tensorboard 命令

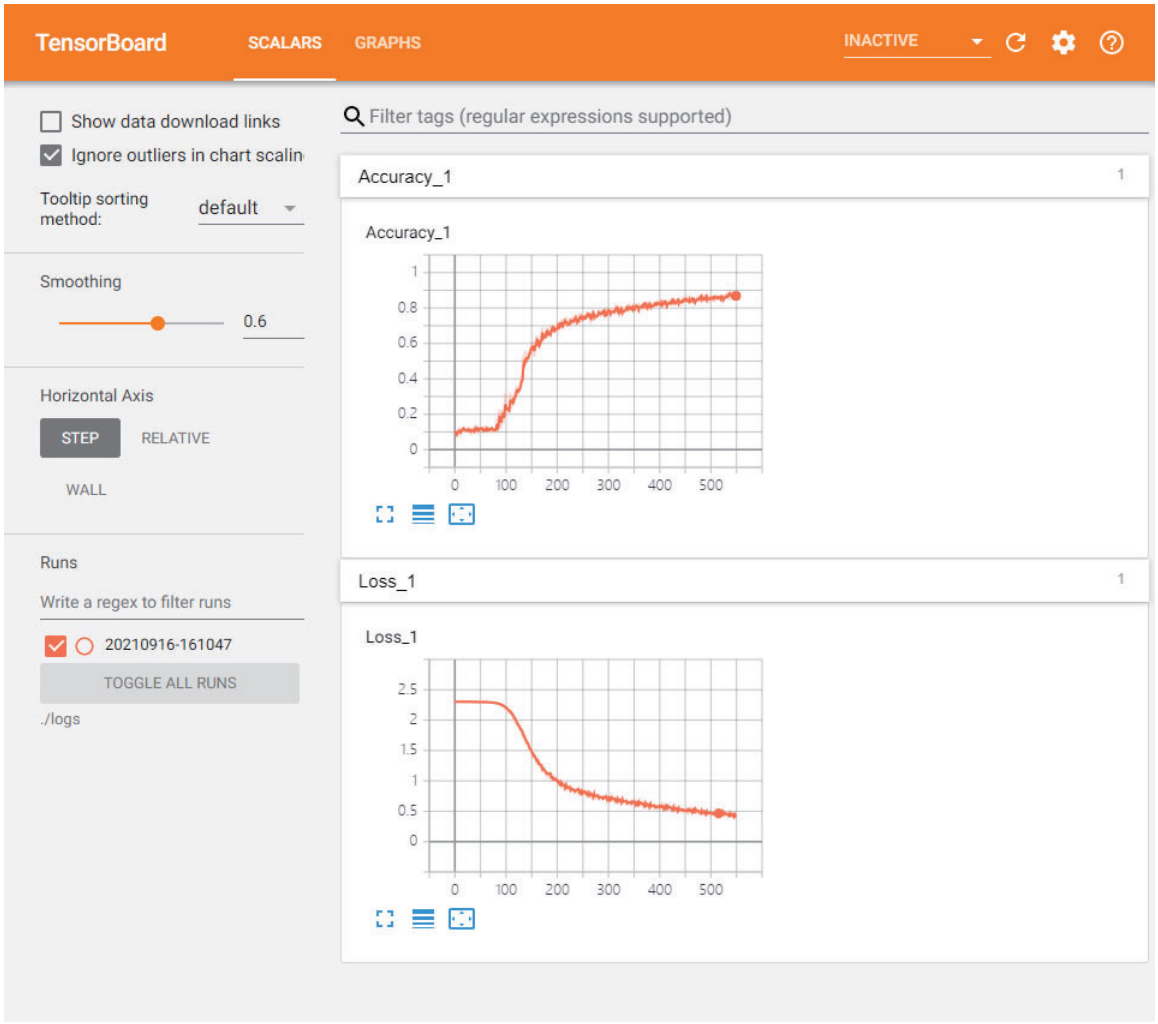

图 3: Tensorboard 页面