第3章 TensorFlow机器学习框架

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | 人工智能基础与应用 | | 章名 | | | TensorFlow机器学习框架 | |
| 教学内容 | TensorFlow机器学习框架 | | | | 课时 | 2 | |
| 项目性质 | □演示性 □验证性 □设计性 √综合性 | | | | | | |
| 授课班级 |  | 授课日期 | |  | | 授课地点 |  |
| 教学目标 | 了解TensorFlow机器学习框架  掌握TensorFlow环境搭建  掌握TensorFlow计算机加速 | | | | | | |
| 教学内容 | 1. TensorFlow介绍 2. TensorFlow环境搭建 3. TensorFlow计算机加速 | | | | | | |
| 教学重点 | TensorFlow环境搭建、TensorFlow计算机加速 | | | | | | |
| 教学难点 | TensorFlow环境搭建、TensorFlow计算机加速 | | | | | | |
| 教学准备 | 装有Python的计算机  教学课件PPT  教材：《人工智能基础与应用（微课版）》 | | | | | | |
| 作业设计 |  | | | | | | |

教学过程

|  |  |
| --- | --- |
| **教学环节** | **教学内容与过程**  **（教学内容、教学方法、组织形式、教学手段）** |
| **课前组织** | 做好上课前的各项准备工作（打开计算机、打开课件、打开软件、打开授课计划、教案等），吸引学生注意力。 |
| **课程说明** | 【课前说明】  通过TensorFlow进行图像风格迁移引入本节课学习内容，先介绍TensorFlow的计算图（Graph）、张量（Tensor）和会话（Session）等基本概念。  【目的】  使学生从了解本节课的学习目标、学习重点、考评方式等方面明确课程学习的要求和目标。 |
| **课程内容描述** | 3.1 TensorFlow介绍  2011年，Google推出了人工智能深度学习系统DistBelief。基于这个系统，Google能够扫描数据中心数以千计的核心，并建立更大的神经网络。DistBelief系统将Google应用中的语音识别成功率提高了25%，该系统还在Google Photos中建立了图片搜索，并驱动了Google的图片字幕匹配实验。但是由于DistBelief和Google内部的基础设施联系过于紧密，导致几乎不可能分享研究代码。  之后，Google的科学家在DistBelief的代码库上进行了简化和重构，使其变成一个更快、更健壮的应用级别代码库，形成了TensorFlow。2015年11月9日，Google基于Apache 2.0许可开源了TensorFlow，其迅速成为最受欢迎的机器学习开源框架之一，并构建起庞大的开发者生态。TensorFlow是一个开源框架，或者称为开源工具。  思考一下如何识别一幅图像，如识别一只狗的图像。首先需要提取出该类图像的特征，在机器学习得到发展之前，工程师需要懂得图像领域非常专业的知识，才可以更好地提取图像特征。但是在机器学习以及深度学习得到发展之后，计算机可以通过神经网络自己提取相应特征，人为提取特征的工作量减少，TensorFlow等开源框架就应运而生了。 3.1.1 TensorFlow基础介绍 TensorFlow可以做很多有趣的工作，如图像风格迁移，通过神经网络可以将一幅图片的风格迁移到另一幅图片上。下图所示是荷兰后印象派画家梵·高的The Starry Night（《星月夜》），将其作为被迁移的图片。    被迁移的图片  原始图片为麻省理工学院Stata中心，如图所示。    原始图片  迁移后的图片如图所示。    迁移后的图片  自2015年11月发布起，TensorFlow经历了多次的版本变化，如表所示。  TensorFlow版本变化   |  |  |  | | --- | --- | --- | | 时间 | TensorFlow版本 | 主要变化 | | 2015年12月 | V0.6 | TensorFlow支持GPU加速 | | 2016年12月 | V0.12 | TensorFlow支持Windows | | 2017年2月 | V1.0 | 加入和改进了一些高级API，包括Keras | | 2017年11月 | V1.5 | 增加了动态图机制和用于移动端的轻量级TensorFlow Lite版本 | | 2018年5月 | V1.6 | 增加了支持Cloud TPU、模型和pipeline功能 | | 2018年6月 | V1.8 | 增加了TensorFlow的分布式训练功能 | | 2019年3月 | V2.0 Alpha | 使用Keras和Eager Execution可以轻松建立简单的模型并执行，通过清除不推荐使用的API和减少重复来简化API |   为了让读者更好地理解TensorFlow，下面从3个不同的角度对其进行分析。  （1）TensorFlow的计算模型为计算图（Graph），TensorFlow的名字本身由两个单词构成，即Tensor和Flow。Tensor指的是张量，在TensorFlow中，Tensor可以简单地理解为多维数组，而Flow翻译过来是“流”，表达了张量之间通过计算进行相互转换的含义。在TensorFlow中，每一个运算都是一个节点，在整体代码中，系统会维护一个默认的计算图。  （2）TensorFlow的数据模型为张量（Tensor），在TensorFlow中，所有的数据类型都表示为张量。张量是一个多维数组，如果直接打印某一个张量，并不会像打印List或NumPy一样输出它的值，而是会得到一个结构，结构中包括该张量的名称、维度和类型。  （3）TensorFlow的运行模型为会话（Session），在TensorFlow中，有了数据模型以及计算模型后，在代码执行过程中需要使用会话，会话负责管理代码运行时的所有资源。如果没有指定，会话将会自动加入系统生成的默认计算图中，执行其中的运算。 3.1.2 分布式TensorFlow 如果一台计算机上有很多GPU，那么通过GPU并行运算的方式可以得到很好的加速效果。但是一台计算机可携带的GPU毕竟有限，要想进一步提升速度，可以将TensorFlow分布地运行在多台机器上。2016年4月14日，Google发布了分布式TensorFlow。  分布式TensorFlow有一些基本概念，下面对其进行介绍。  （1）task：一个task一般会关联到某个单一的TensorFlow服务端的处理过程，属于一个特定的Job，并且在该Job的任务列表中有唯一的索引，可以将其理解为每台机器上的一个进程。  （2）Job：Job分为ps、worker两种，一个Job包含一系列致力于某个相同目标的task。例如，一个叫ps的Job会处理存储与更新变量相关的工作，而一个叫worker的Job会承载那些用于计算密集型任务的无状态节点。一般来说，一个Job中的task会运行在不同的机器中。  （3）Cluster（集群）：一个TensorFlow集群包含一个或者多个TensorFlow服务端，集群被切分为一系列Job，而每个Job又会负责一系列的task。一个集群会专注于一个相对高层的目标，如用多台机器并行地训练一个神经网络。  使用分布式TensorFlow训练深度学习模型有两种方式。第一种是计算图内分布式（In-graph模式），在该模式中，所有的计算任务使用同一个计算图中的变量。第二种是计算图间分布式（Between-graph模式），在该模式下，数据并行，每台机器使用完全相同的计算图，每个计算图都是独立的，但数据同步比较困难。  3.2 TensorFlow环境搭建  TensorFlow框架的环境搭建是学习TensorFlow的基础，了解并亲自动手搭建环境可以使使用者加深对TensorFlow的认识。TensorFlow版本、适用平台众多，本节主要介绍本书用到的环境以及后续开发过程中常用的环境。 3.2.1 安装Anaconda 第2章介绍了直接在计算机上安装Python 3解释器的方法。Python 2和Python 3的解释器是无法兼容的，在使用Python 3时需要将计算机上的环境设置为Python 3解释器的环境，如果使用Python 2来执行一些代码，就需要更换本地环境为Python 2解释器的环境，当然Python 3不同版本的API也有差别，所以使用直接搭建的方式在一台计算机上让不同版本的Python解释器兼容是无法实现的。  Anaconda是一个开源的包和环境的管理器，用于在同一台机器上管理不同版本的软件包及其依赖，并能够在不同的环境之间切换。其基于云的存储库，可以查找并安装Conda、Python等7500多个科学包及其依赖项。  本书介绍Windows版本的Anaconda软件的安装。对于Linux版本的Anaconda，读者可自行安装。  下载64位Anaconda，然后双击下载好的包进入安装界面，单击“Next >”按钮，如图所示。    安装界面  进入同意许可（License Agreement）界面，单击“I Agree”按钮同意协议许可，如图所示。    同意许可界面  进入选择安装类型（Select Installation Type）界面，选中“All Users (requires admin privileges)”单选按钮，为计算机所有用户进行安装，单击“Next >”按钮，如图所示。    选择安装类型界面  进入选择安装路径（Choose Install Location）界面，单击“Browse”按钮选择安装路径，然后单击“Next >”按钮，如图所示。    选择安装路径界面  进入高级安装选项（Advanced Installation Options）界面，勾选“Add Anaconda to the system PATH environment variable”“Register Anaconda as the system Python 3.7”复选框，将Anaconda添加到系统环境变量，并默认使用Python 3.7版本，单击“Install”按钮，如图所示。    高级安装选项界面  进入安装完成（Installation Complete）界面，等待安装完成，这个过程比较长，安装完成后单击“Next >”按钮，如图所示。    安装完成界面  进入软件推荐界面，Anaconda推荐使用PyCharm作为IDE，PyCharm的安装在第2章介绍过，单击“Next >”按钮，如图所示。  进入感谢安装Anaconda界面，取消勾选“Learn more about Anaconda Cloud”“Learn how to get started with Anaconda”复选框，相关知识将在本书后续内容中进行讲解，单击“Finish”按钮，完成安装，如图所示。    软件推荐界面    感谢安装Anaconda界面  安装完成后，打开命令提示符窗口，输入“conda -V”以获得Anaconda版本号，如果输出正常，证明Anaconda正确安装，如图所示。    Anaconda正确安装  找到并打开Anaconda Navigator软件，进入Anaconda界面模式，Anaconda Navigator图标如图所示。    Anaconda Navigator图标  Anaconda在默认情况下只有一个环境“base(root)”，新建TensorFlow的环境进行学习，在主界面中选择“Environments”→“Create”目录，弹出“Create new environment”对话框，选择Python版本为3.6，在“Name”文本框中输入“TensorFlow”，单击“Create”按钮完成环境创建，如图所示。    “Create new environment”对话框  等待创建完成，创建完成的界面如图所示。    创建完成的界面  新环境创建完成后，在命令提示符窗口输入“activate TensorFlow”，进入新环境，如图所示。    通过输入命令进入新环境 3.2.2 使用pip的Windows环境安装 在Anaconda的TensorFlow环境下，可以使用conda命令或者pip命令安装NumPy、Pandas和Matplotlib库，然后安装TensorFlow。TensorFlow有CPU和GPU两种版本，下面分别介绍这两种版本的安装方式。  1．CPU版本安装  在Anaconda的TensorFlow环境下的命令提示符窗口内输入“pip install tensorflow==1.9.0”，安装1.9.0版本的TensorFlow及其相关依赖包（安装过程需全程联网），如图所示。    安装CPU版本的TensorFlow及相关依赖包  安装完成后测试是否安装成功，在Anaconda的TensorFlow环境下的命令提示符窗口内输入“python”，进入Python环境后输入“import tensorflow as tf”命令，查看是否报错，不报错说明安装成功，如图所示。    查看TensorFlow是否安装成功  2．GPU版本安装  安装GPU版本的TensorFlow之前需要查看自己的计算机或嵌入式设备是否支持该版本。如果支持GPU版本的TensorFlow，在安装之前需要安装CUDA和cuDNN。要安装cuDNN，需要注册成为NVIDIA用户，填写问卷调查才可以下载。在安装时需注意CUDA、cuDNN的版本匹配问题，安装成功后，即可开始安装GPU版本的TensorFlow，可以新建一个Anaconda环境安装GPU版本的TensorFlow。在Anaconda环境下的命令提示符窗口输入“pip install tensorflow-gpu==1.9.0”命令即可完成安装。 3.2.3 使用pip的Linux环境安装 在Linux（以Ubuntu为例）下安装Anaconda和在Windows下安装的过程一样，在Linux系统下安装完Python（很多Linux系统自带Python 2和Python 3）和pip工具后，可以直接输入“pip3 install tensorflow ==1.9.0”命令完成CPU版本TensorFlow的安装，如图3-19所示。    在Ubuntu下安装TensorFlow 3.2.4 使用源代码编译安装 在Linux操作系统上通过源码安装TensorFlow，需要使用Bazel编译工具。安装依赖JDK8之后通过apt-get安装或者下载Bazel源码，然后安装Bazel编译工具。Bazel编译工具安装完成后，下载TensorFlow源码目录。进入TensorFlow的源码目录后，输入“./configure”命令进行TensorFlow编译安装配置，配置过程中会出现Python路径询问等问题，可以根据自己的实际情况进行选择或者选择默认配置。在配置完成后，采用Bazel命令安装TensorFlow，输入“bazel build -c opt /tensorflow/tools/pip\_package:build\_pip\_package”。安装完成后输入“bazel-bin/tensorflow/tools/pip\_package/build\_pip\_package/tmp/tensorflow\_pkg”命令，在tmp/tensorflow\_pkg目录下生成扩展名为.whl的文件，然后使用pip3命令安装该文件即可。  3.3 TensorFlow计算机加速  在搭建网络训练模型时，由于网络中的参数众多、运算量大，所以训练过程比较缓慢，如果计算机不具有CPU，可以基于CPU版本的TensorFlow使用更高级的指令集，如SSE、AVX，加速训练过程；如果计算机支持GPU，可以使用GPU加速训练过程。 3.3.1 TensorFlow的使用 使用CPU版本的TensorFlow完成向量加法运算，举例介绍如下。  【例3-1】 新建TensorFlow目录，在TensorFlow目录下新建文件，命名为test.py，在PyCharm中编写代码实现向量的加法运算。  # tf.constant是一个计算，计算结果是一个张量，保存在变量a或者b中  a = tf.constant([1.0, 2.0], name="a")  b = tf.constant([3.0, 4.0], name="b")  # 将a和b相加，相加后的名字为“add”  result = tf.add(a, b, name = "add")  # 输出  print(result)  # 创建一个会话，通过Python上下文管理器来管理该会话  # 启动默认图表  with tf.Session() as sess:  print("a = [1.0, 2.0], b = [3.0, 4.0]")  print("两个向量相加: a + b = ", sess.run(result))  # 将数据写到日志中  summary\_writer = tf.summary.FileWriter("log", sess.graph)  代码中的“print(result)”会输出：  Tensor("add:0", shape=(2,), dtype=float32)  输出的张量有3个属性。第1个属性是名称，它不仅仅是该张量的唯一标识符，还可以表示该张量是如何计算出来的。第2个属性是张量的维度，“shape=(2,)”表示一个一维数组，长度为2。第3个属性是类型。每个张量都有自己的类型，如果两个张量在运算时类型不匹配，运算会报错。  本例中使用“with tf.Session() as sess”创建一个会话，创建的会话在执行完成后会自动关闭和释放资源。如果采用“sess = tf.Session()”的方式创建会话，需要使用sess.close()函数手动关闭资源和进行资源回收。无论是哪种创建会话的方式，在创建会话的时候都会关联默认图。  本例使用“summary\_writer = tf.summary.FileWriter("log", sess.graph)”语句对数据进行记录，“log”为日志所在的位置，“sess.graph”为TensorFlow代码中的图。  执行完毕后会输出两者的和：  a = [1.0, 2.0], b = [3.0, 4.0]  两个向量相加: a + b = [4. 6.]  同时，执行完毕后也会在TensorFlow目录下生成一个名为log的目录。在TensorFlow目录下进入命令提示符窗口，通过TensorBoard库可以看到计算图，在Anaconda的TensorFlow环境下输入“tensorboard--logdir=log”，会得到一个HTTP链接，命令提示符界面如图所示。    使用命令得到HTTP链接  使用浏览器进入该链接后，可以看到两个向量相加的计算图，如图所示。    计算图 3.3.2 TensorFlow使用GPU加速 3.3.1小节的示例使用CPU版本的TensorFlow完成两个向量的相加，本小节将展示如何使用TensorFlow进行单个GPU的加速计算。  如果计算机上只安装GPU版本的TensorFlow，那么3.3.1小节的示例会直接调用GPU版本的TensorFlow执行运算。也可以通过tf.device()函数指定设备进行运算，CPU在TensorFlow中被命名为“/cpu:0”。在计算机中，即使有多个CPU，TensorFlow也不会区分，名称始终为“/cpu:0”。如果一台计算机上有多个GPU，那么第一个GPU会被命名为“/gpu:0”，第二个GPU会被命名为“/gpu:1”，以此类推。  【例3-2】 在TensorFlow项目下新建test\_gpu.py文件，使用GPU版本的TensorFlow实现向量的相加（在同一个环境下可同时安装CPU版本的TensorFlow和GPU版本的TensorFlow）。  import tensorflow as tf  # 通过tf.device()将运算指定到CPU上  with tf.device("/cpu:0"):  a = tf.constant([1.0, 2.0], name="a")  b = tf.constant([3.0, 4.0], name="b")  # 通过tf.device()将运算指定到GPU上  with tf.device("/gpu:0"):  result = tf.add(a, b, name = "add")  # 利用log\_device\_placement将参与运算的设备输出  sess = tf.Session(config=tf.ConfigProto(log\_device\_placement = True))  print(sess.run(result))  通过例3-2的代码可知，并不是所有的操作都放在GPU上，a = tf.constant([1.0, 2.0], name="a")和b = tf.constant([3.0, 4.0], name="b")两个定义a和b常量的操作会通过“with tf.device("/cpu:0")”函数加载到CPU上，使用result = tf.add(a, b, name = "add")做相加运算时，这个过程会被加载到GPU上，并且这个操作会将参与运算的设备信息打印出来，在GTX1050上得到如下运算结果。  Device mapping:  /job:localhost/replica:0/task:0/device:GPU:0 -> device: 0, name: GeForce GTX 1050, pci bus id: 0000:01:00.0, compute capability: 6.1  add: (Add): /job:localhost/replica:0/task:0/device:GPU:0  a: (Const): /job:localhost/replica:0/task:0/device:CPU:0  b: (Const): /job:localhost/replica:0/task:0/device:CPU:0  [4. 6.] |
| **总结评价** | 本节课主要介绍TensorFlow的一些基础概念与相关操作。 |