花卉识别

XXXXXXXX（引言）。（**本文档中已经设置好同学们需要使用的样式，以及图片、代码的格式，所以同学们在编写项目报告时，需要直接在这份文档中进行编写**）

**学习目标**

1. XXXXXXXX。
2. XXXXXXXX。
3. ……

# 设计目的及任务描述

## 任务描述

XXXXXXXX（为什么做这个任务，以及这个任务的总目标）。

## 任务分析

1. XXXXXXXX。
2. XXXXXXXX。
3. ……

## 设计目的

XXXXXXXX（正文）。

# 软硬件配置及环境搭建

## 任务描述

软硬件配置及环境搭建是构建神经网络模型以及模型训练前重要的一环，在此任务中，需要搭建好项目的环境，为随后的模型搭建及训练做好准备。

## 任务分析

1. 利用anaconda搭建python虚拟环境。
2. 安装tensorflow。

## 软硬件配置

硬件：

12核11th Gen Inter® Core™ i5-11400H @ 2.70GHz

NVIDIA GeForce RTX 3060 Laptop GPU

软件：

Python 3.7

Anaconda3 2022.10(Python 3.9.1364-bit)

PyCharm Community Edition 2022.3

TensorFlow2.1

## 环境搭建

首先，在Anaconda官网：https://www.anaconda.com/下载Anaconda3 社区版。需要将Anaconda加入系统环境变量，以便于与其它软件合作编程。

安装好 Anaconda后，便需要创建一个conda虚拟环境，以便同一个机器上安装不同版本的软件包及其依赖。使用anaconda创建一个名为TF2.1使用python3.7的虚拟环境。

创建好环境后，需要进入虚拟环境安装所必须的一些支持库。

首先，进入到已经创建好的虚拟环境。

安装英伟达的SDK（可以使用GPU加速），使用tensflow2.1，选择cudatoolkit10.1版本。注意这两者有严格的版本要求，一定要安装合适的版本，否者后续会无法启动 GPU 加速或者直接安装失败。

同理，安装英伟达深度学习软件包（7.6 版本）

最后，安装 tensflow2.1

由于先前已经将Anaconda添加至系统环境变量，因此只需要在使用PyCharm创建项目的时候指定该虚拟环境即可。

环境配好了之后，我们便可以使用PyCharm进行TensorFlow的相关操作了。

使用代码2-1进行测试：

代码 2‑1 测试环境是否配置正常命令

|  |
| --- |
| import tensorflow as tf  tensorflow\_version=tf.\_\_version\_\_7  gpu\_available=tf.test.is\_gpu\_available()  print("tensorflowversion:",tensorflow\_version,"\tGPUavailable:",gpu\_available) |

在控制台中输出有：tensorflow version: 2.1.0 GPU available: True并且程序没有 error 即可表明环境配置成功

# 花卉数据预处理

## 任务描述

在进行模型训练前，需要先爬取足够多的图片数据，同时由于爬取的图片可能包含水印，会对模型训练造成干扰，因此这些图片也需要进行去水印处理，最后需要根据图片中花的类型划分数据集，为模型训练做准备。

## 任务分析

1. 爬取七种花的图片。
2. 对图片进行去水印处理。
3. 根据花的类型划分数据集。

## 爬取图片

首先需要通过爬虫来获取七种类型的花的图片，由于各种类型的花图片可以直接在图片网站上搜得，因此可以通过指定七个URL（每种类型的花对于一个URL），在该URL下爬取对应类型的花的图片。具体处理过程如下：

（1）定义一个包含URL和下载文件路径的字典，其中包括七种不同的花卉类型的图片网址。

（2）通过使用requests库和get方法来获取每个URL指定的网页的HTML源代码。

（3）使用正则表达式从每个页面中查找匹配的图像链接URL。

（4）通过使用requests库中的get方法获取这些URL链接指定的图像文件。

（5）将图像文件保存到指定的文件路径中。

爬虫代码如代码 3‑1所示:

代码 3‑1 爬取图片代码

|  |
| --- |
| reg = re.compile(pattern)  for url in urls:  # 查看图片保存的路径是否存在，若不存在则新建目录  if not os.path.exists(suffix + url):  os.mkdir(suffix + url)  page = requests.get(urls[url]).text  links = re.findall(reg, page) # 匹配网页中所有的url  links\_ = [] # 清洗爬取的url  for link in links:  links\_.append(link[1:-1]) # 去除url两边的引号  num = 0  pos = suffix + url + '/'  for i in links\_: # 保存图片  a = requests.get(i)  with open(pos + '%s.jpg' % num, 'wb') as f:  f.write(a.content)  num += 1  print(url + '爬取完毕！') |

程序执行完成后，将会在指定路径下生成七个文件夹（分别命名为：鸡蛋花、玫瑰花、牡丹、蒲公英、牵牛花、向日葵、郁金香），如图 3‑1所示。每个文件夹内包含有爬取的对应类型的花的图片。

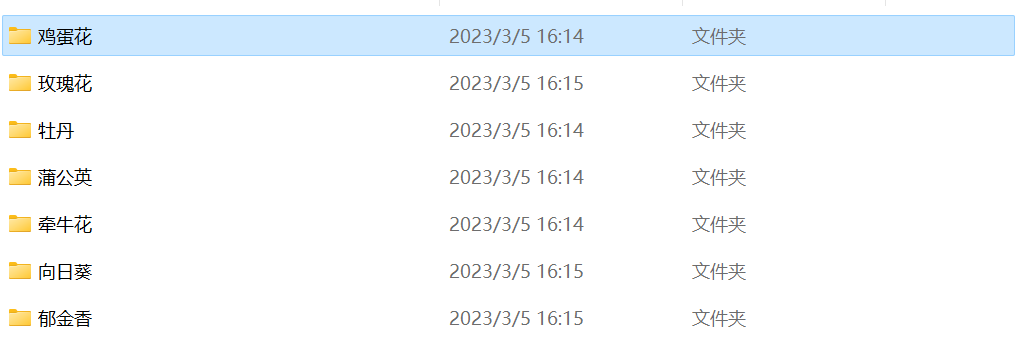


图 3‑1 爬取成功后生成的文件夹

## 图片去水印

XXXXXXXX（正文）。

## 划分数据集

为了评估模型的区分度、稳定性、鲁棒性等特性，程序将样本拆分为：训练集、测试集、验证集。其中，训练集用于训练有监督模型，拟合模型，调整参数，选择入模变量，以及对算法做出其他抉择；测试集，用于评估训练出的模型效果，但不会改变模型的参数及效果，一般验证模型是否过拟合或者欠拟合，决定是否重新训练模型或者选择其他的算法；验证集，因为训练集和测试集均源自同一分布中，随着时间的流逝，近期样本的分布与训练模型的样本分布会有变化，需要校验训练好的模型在近期样本是否有同样的效果，即模型的稳定性、鲁棒性、泛化误差。

程序具体处理过程如下：

（1）在路径下创建三个文件夹，分别命名为：train、test、val（训练集、测试集、验证集）。

（2）打乱原始数据集，以用户自定义的数据集比例，将各图片分配至各数据集下。

（3）划分完成，输出结果。

划分数据集的核心代码如代码 3‑2所示：

代码 3‑2 划分数据集核心代码

|  |
| --- |
| for i in current\_data\_index\_list:  src\_img\_path = os.path.join(current\_class\_data\_path, current\_all\_data[i])  if current\_idx <= train\_stop\_flag:  copy2(src\_img\_path, train\_folder)  train\_num = train\_num + 1  elif (current\_idx > train\_stop\_flag) and (current\_idx <= val\_stop\_flag):  copy2(src\_img\_path, val\_folder)  val\_num = val\_num + 1  else:  copy2(src\_img\_path, test\_folder)  test\_num = test\_num + 1 |

数据集完成划分后产生的文件如图 3‑2所示：

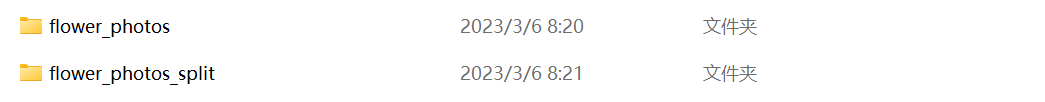


图 3‑2 划分后的文件

其中，flower\_photos为原数据集文件夹，flower\_photos\_split为划分后的数据集文件夹，其内的文件如图 3‑3所示：

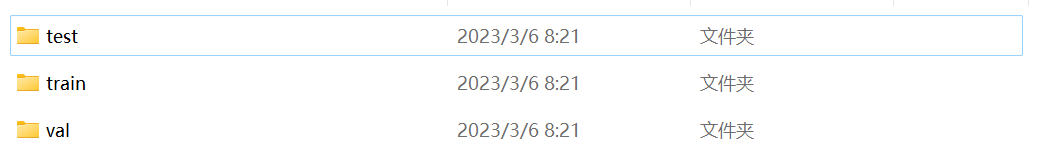


图 3‑3 flower\_photos\_split内的文件

可以看出，程序已经按用户设定好的比例将数据集划分为训练集(train)，测试集(test)，验证集(val)。

## 图像增强

XXXXXXXX（正文）。

# 使用CNN卷积神经网络训练模型

## 任务描述

XXXXXXXX（为什么做这个任务，以及这个任务的总目标）。

## 任务分析

1. XXXXXXXX。
2. XXXXXXXX。
3. ……

## 了解CNN卷积神经网络

XXXXXXXX（正文）。

## 神经网络结构

XXXXXXXX（正文）。

## 测试结果

XXXXXXXX（正文）。

# Streamlit前端设计

## 任务描述

XXXXXXXX（为什么做这个任务，以及这个任务的总目标）。

## 任务分析

1. XXXXXXXX。
2. XXXXXXXX。
3. ……

# 小结

XXXXXXXX（正文）。

# 报告格式

**以下内容为编写项目报告是的格式要求，仅供参考。在具体的编写时，需将这部分内容删除。**

## 格式

1. 编写时，必须直接使用本文档，因为设置的样式可以直接使用，不需要自己手动设置，常用到的样式如图 1‑1所示。

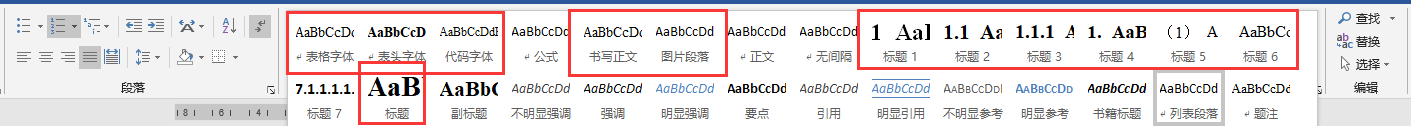


图 7‑1 可用样式

1. 标题使用的顺序严格按照表 1‑1所示的顺序（标题设置比较麻烦，可直接使用本文档）。且要注意，**全文不能出现项目符号**。

表 1‑1 标题、编号级别顺序

|  |  |  |
| --- | --- | --- |
| 标题级别 | 格式 | 对应的样式 |
| 首级 |  | 标题 |
| 一级 | 1 | 标题1 |
| 二级 | 1.1 | 标题2 |
| 三级 | 1.1.1 | 标题3 |
| 四级 | 1. | 标题4 |
| 五级 | （1） | 标题5 |
| 六级 | ① | 标题6 |
| 七级 | a. |  |
| 八级 | （a） |  |
| 九级 |  |  |

**标题、编号需要注意以下几点。**

1. 标题的文字最后是没有任何标点符号的，如本文档的标题。
2. 六级标题，需应用“标题6”样式，其编号需手动插入，插入方法：单击在Word文档的“插入”选项卡的“符号”按钮，选择“其他符号”选项；在弹出的对话框中选择图 1‑2所示的“带括号的字母数字”选项。此时插入对应编号的符号即可。



图 7‑2 六级编号插入方法

## 用词

1. 不要使用“你”“我”“他”等人称代词，**必须使用一般的陈述句。**
2. **语句必须书面化**，类似“来”“去”“说”“讲”“演示”“想”“当然”“就可以”等词尽量不要用，这些词比较偏口语。
3. **尽量少使用“上面”“下面”“以上”“以下”“上述”“接下来”**等指代不明确的词语；如果涉及到图、表、代码等，那么最好用交叉引用介绍清楚具体是哪个内容，如本文档对图、表的编号所示；如果涉及到某个小节的内容，可以用“X.X.X小节”的形式。

## 代码

代码的格式可以直接使用图 5‑1所示的**表格样式**，字体**“代码字体”样式**。代码示例如代码 5‑1所示。



图 7‑3 代码样式

代码 7‑1 代码示例

|  |
| --- |
| aaa |

当代码的运行结果为非表格或图片时，可以用下面的1×1表格。同样，这个也有对应的表格样式，如图 5‑2所示。

|  |
| --- |
|  |



图 7‑4 语法格式或输出结果样式

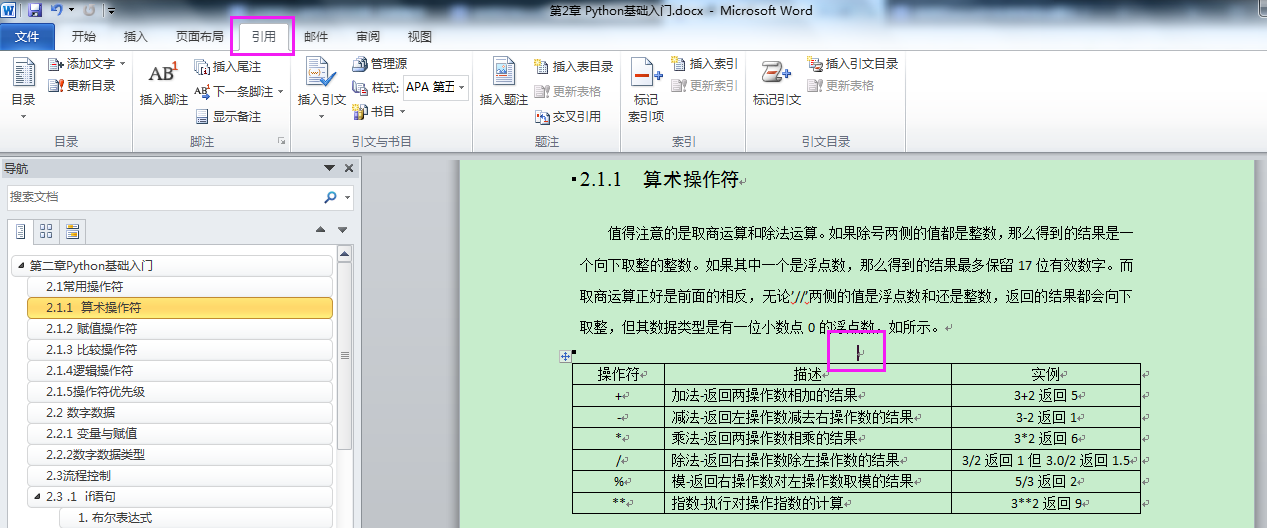
## 图/表/代码的题注

图、表、代码的题注格式如表 1‑2所示。具体示例如本文档的所有图、表、代码。

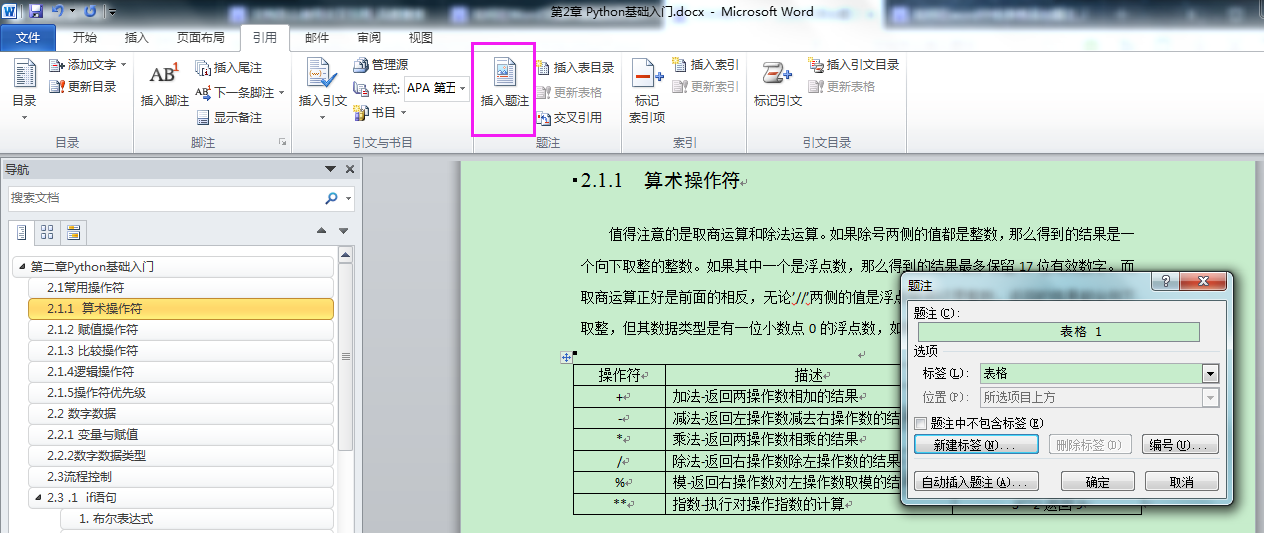
表 1‑2 图、表、代码的题注格式

|  |  |  |  |
| --- | --- | --- | --- |
| 类别 | 题注编号格式 | 引用格式 | 名称位置 |
| 图 | 图 7-1 | 图 7-1 | 下方，居中 |
| 表 | 表 7-1 | 表 7-1 | 上方，居中 |
| 代码 | 代码 7-1 | 代码 7-1 | 上方，居中 |

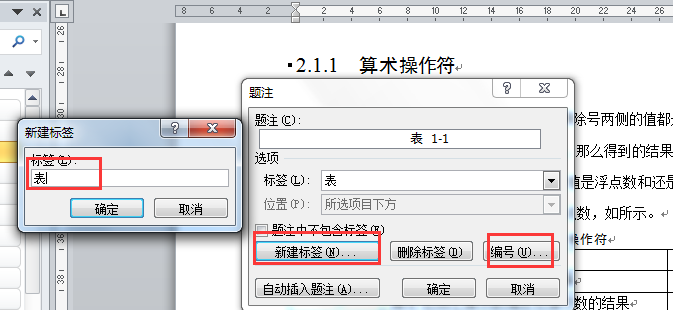
1. 将光标定位在添加题注的位置，将界面切换至“引用”选项卡。



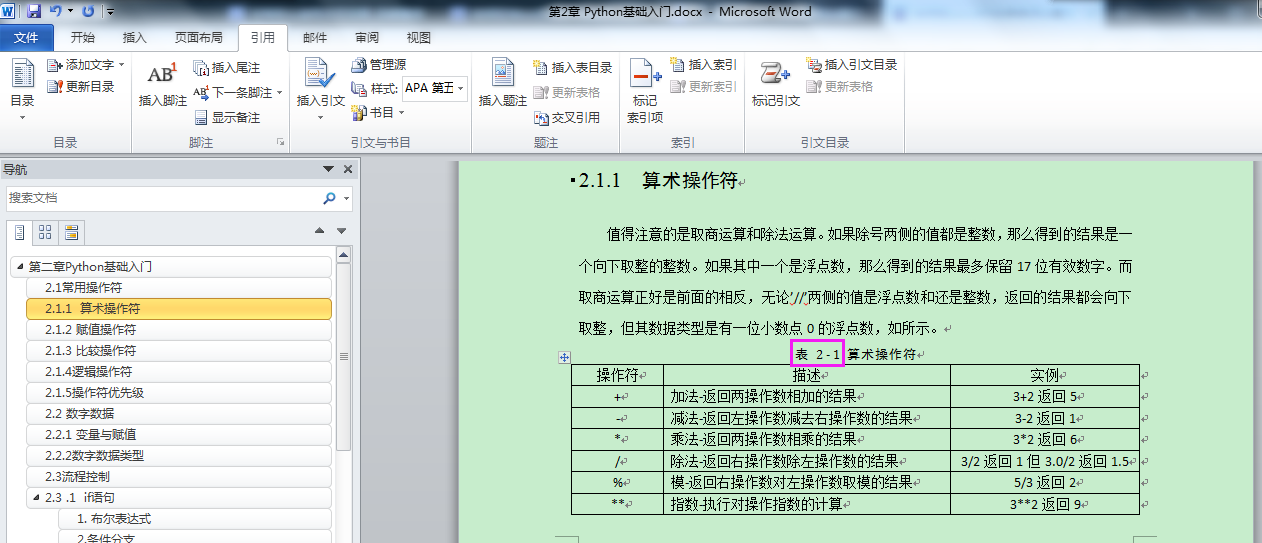
1. 在“题注”组中，单击“插入题注”选项。



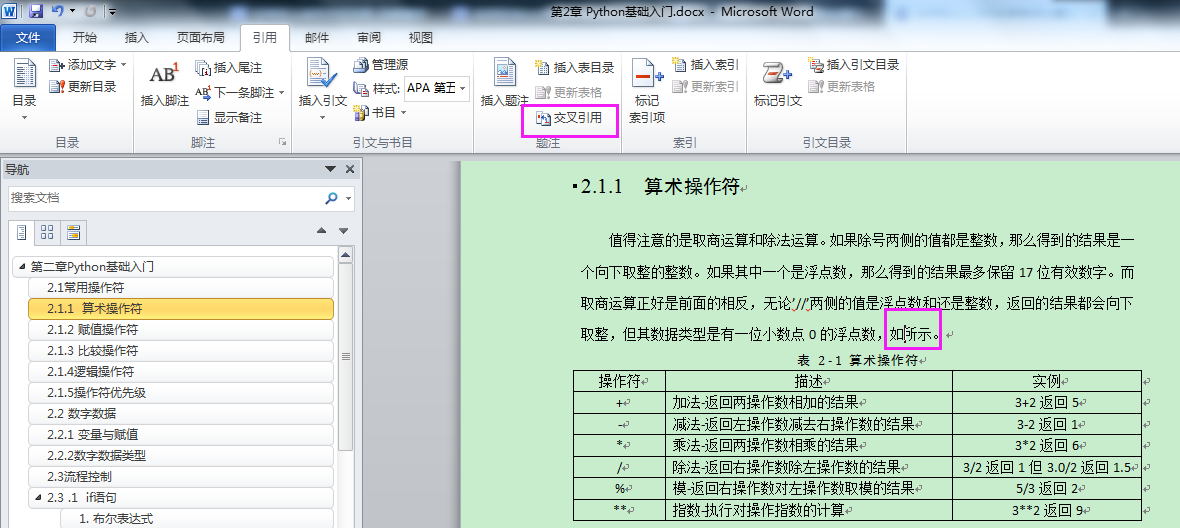
1. 单击“新建标签”后，即可出现如下的设置界面，在“新建标签”对话框中输入需要的题注，如“表”，单击“确定”按钮；单击“编号”，在“包含章节号”前面打钩，单击“确定”按钮。



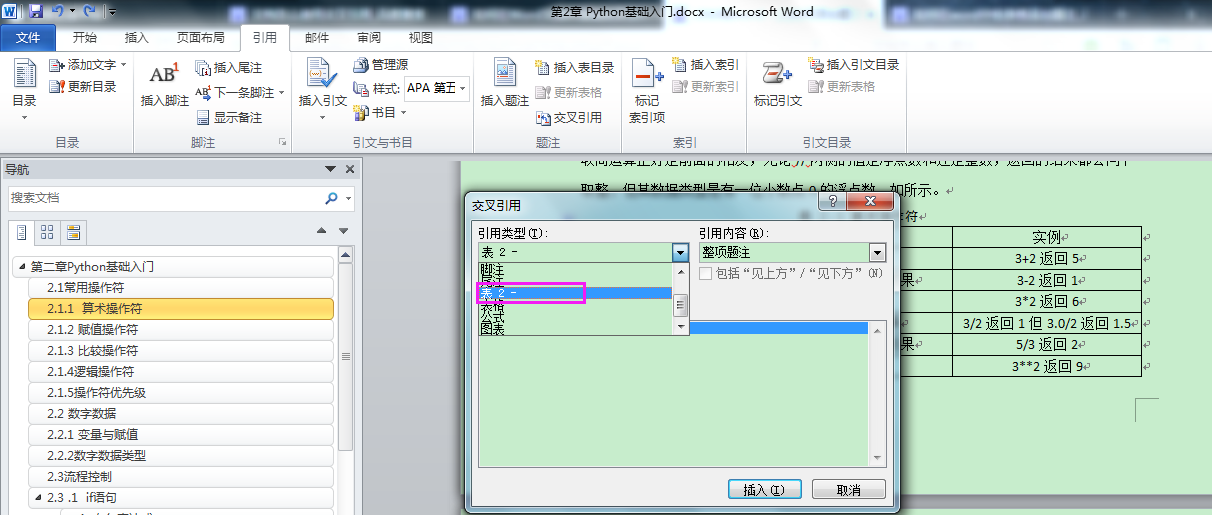
1. 设置好之后，会出现“表 2-1”，在后面写上表名。

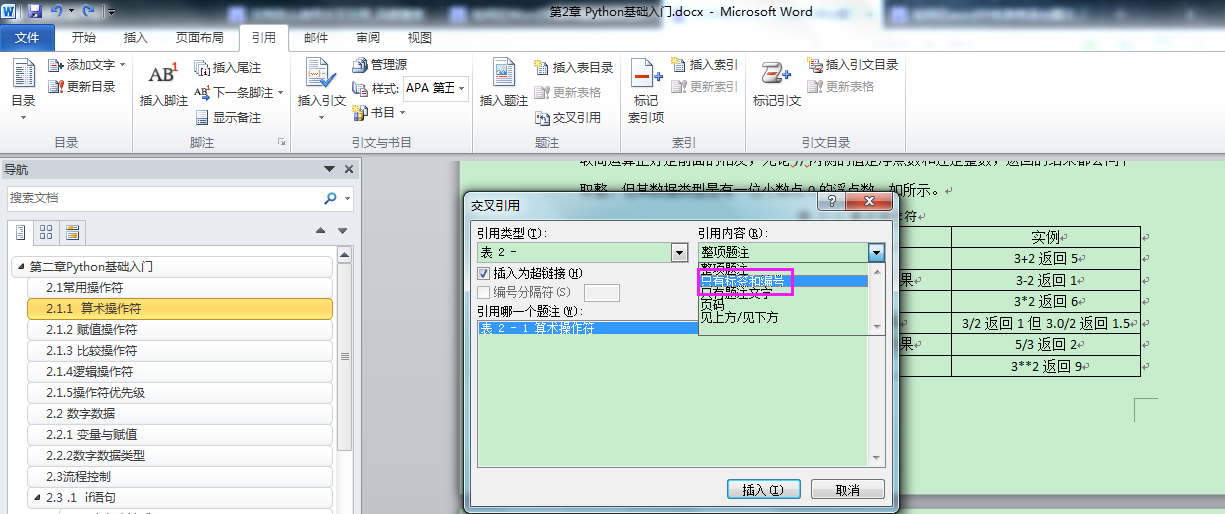


1. 将光标移到需要引用图表的地方，在“引用”选项卡的“题注”组中，单击“交叉引用”选项。

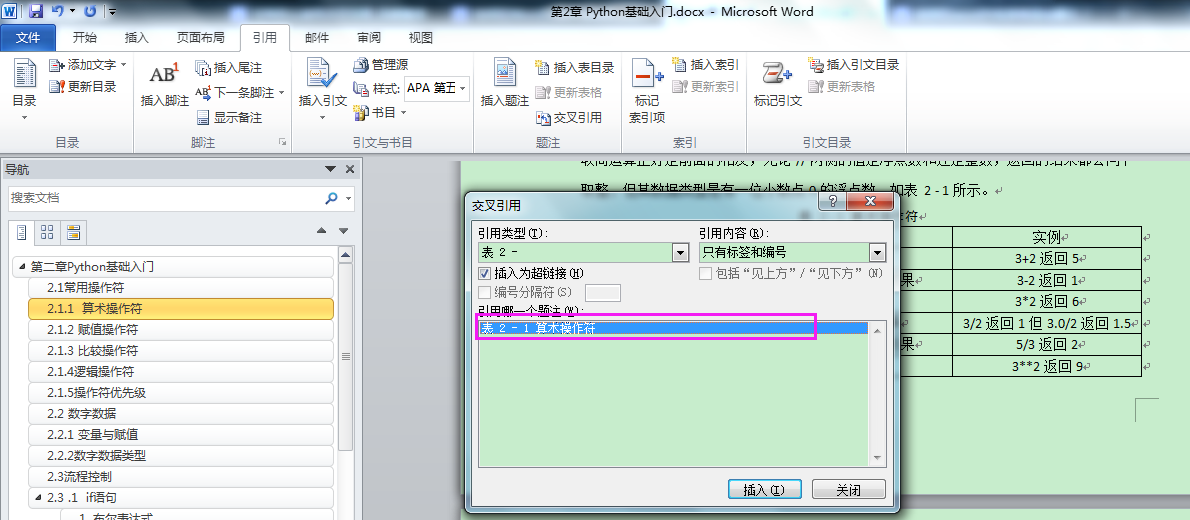


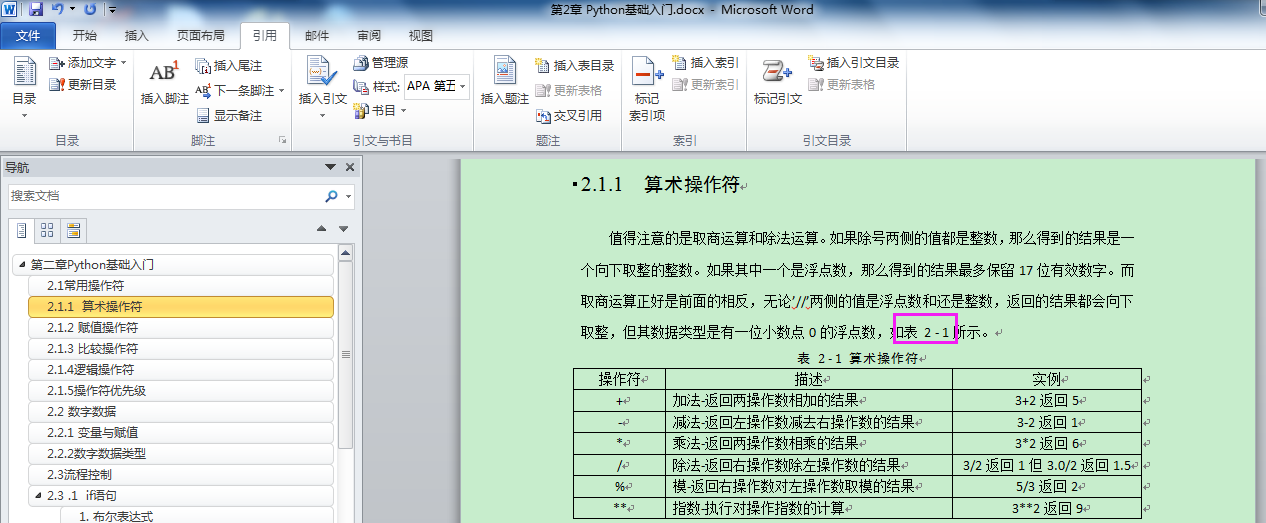
1. 在“引用类型”中选择要插入的标签（在步骤3中可设置），在“引用内容”中选择“只有标签和编号”。





1. 单击要插入的题注，然后单击“插入”按钮即可。





1. 当两个图表中间插入新的图表后，文档中所有的标签会自动更新，但是引用部分的表标签不会变。

此时，选择全文（Ctrl+a），按F9键，即可更新全文的图表引用。

