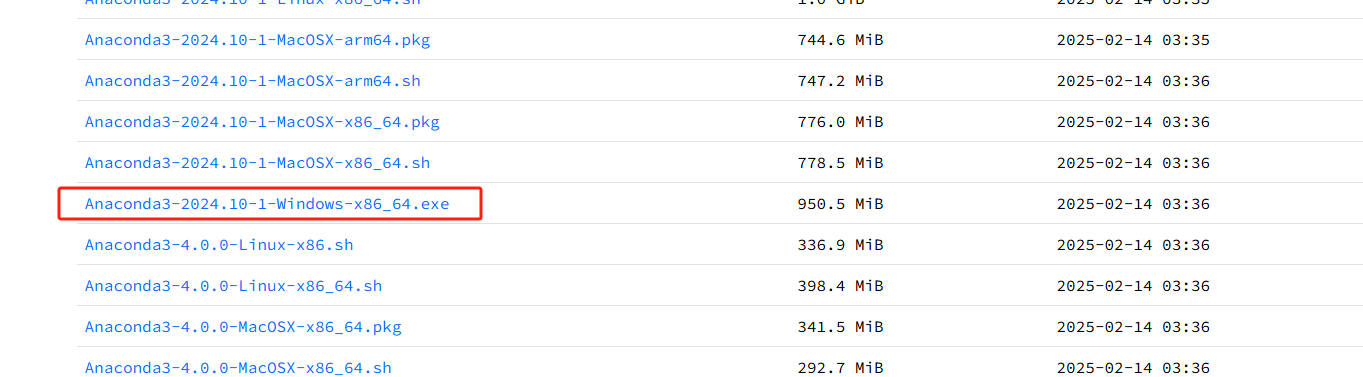
**实验：知识图谱表示学习模型 TransE**

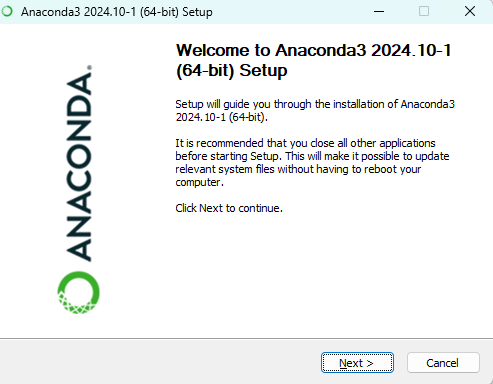
1. 安装Anaconda（已安装跳过此步骤）
2. 下载Anaconda并安装

清华镜像网：

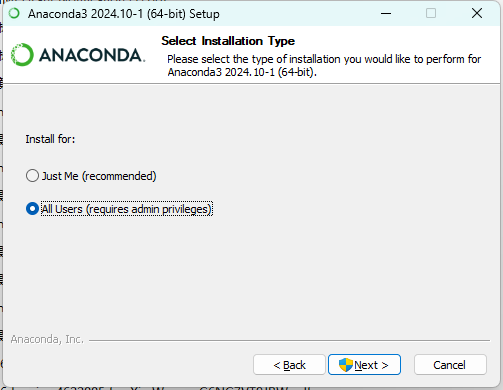
https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/

选择下载最新版本的2024.10-1版本

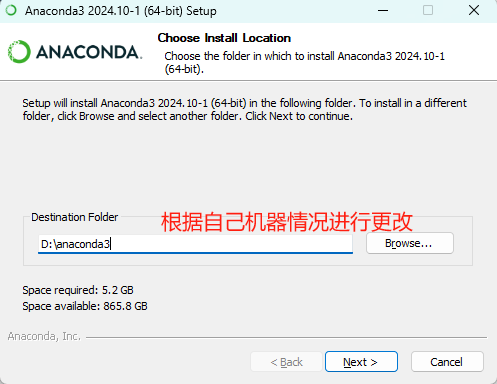


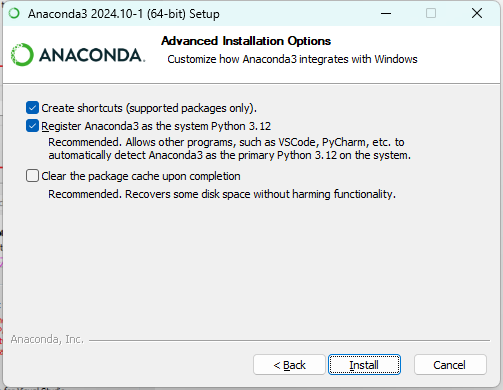


前面我们一直确认next即可。直到这里我们选择all user。

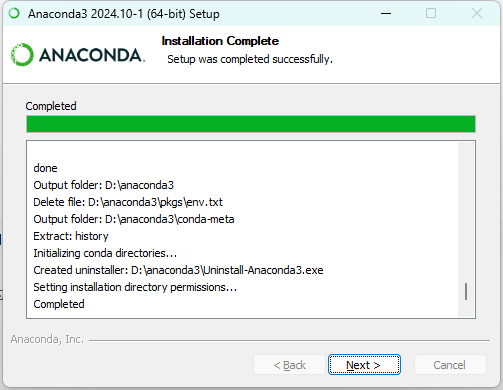


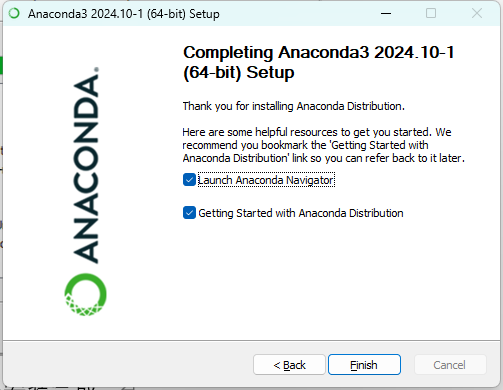
这里选择安装路径，这里最好选择自己的路径（默认安装是安装在C盘）。





然后等待安装即可，安装文件有几GB，时间会可能会比较久，因各自电脑配置而异，耐心等待即可。





到这里anaconda已经安装完成。

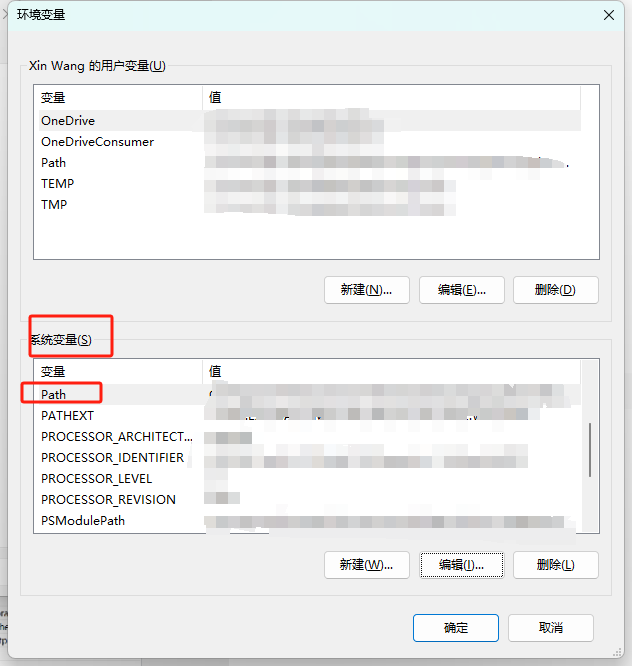
1. 配置环境变量

打开设置，搜索“查看高级系统设置”，点击打开。

若安装时已勾选自动配置， 此步可忽略

打开环境变量。





双击打开，新建。

新建五个变量进去，将下面的五个变量的结合你的anaconda实际安装目录来更改写入。

（这里的anaconda安装路径为D:\anaconda3，把下面的更改为你的anaconda路径即可）

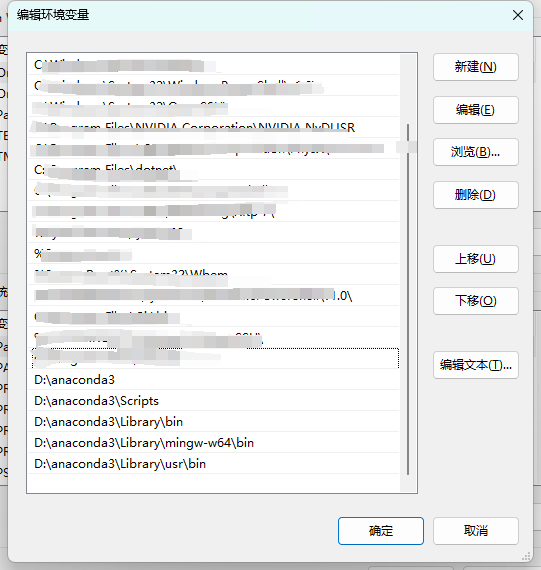
D:\anaconda3

D:\anaconda3\Scripts

D:\anaconda3\Library\bin

D:\anaconda3\Library\mingw-w64\bin

D:\anaconda3\Library\usr\bin

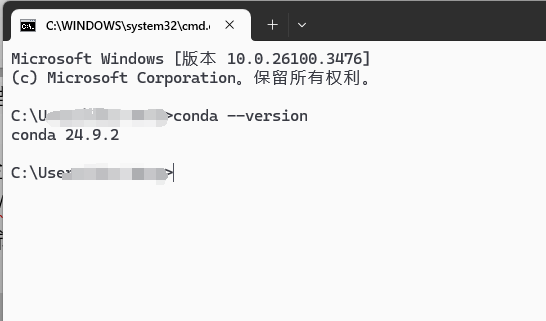


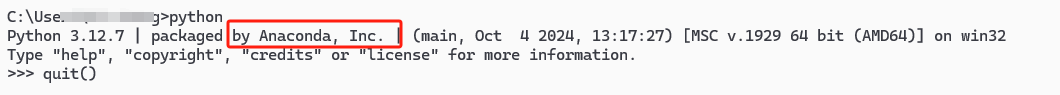
添加结束，确认退出。

1. 检查是否安装成功

按下Win+R，输入cmd打开终端。

输入命令检验。//检验anaconda版本





1. 更改镜像源

（切记，更改镜像源需要在系统环境，如果第四步进入了python环境，需要先输入exit退出，或者重新打开一个终端更改镜像源）

**cmd后依次输入下面命令**  
直接输入以下命令即可，将默认的国外站点更改为国内的镜像源，速度更快！

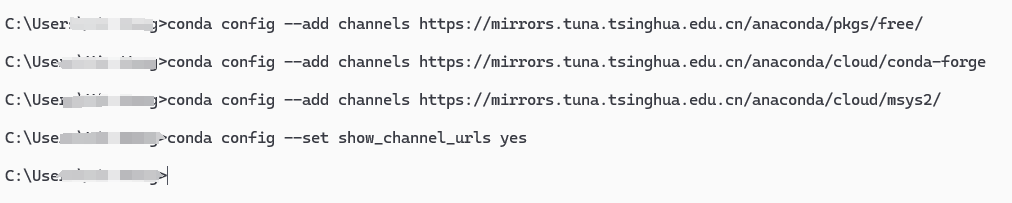
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/

conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge

conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/msys2/

//设置搜索时显示通道地址

conda config --set show\_channel\_urls yes

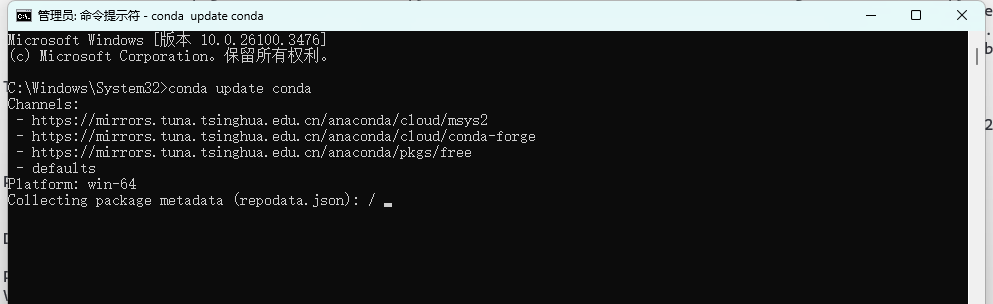


1. 更新包（需要较长时间，实验时建议跳过此步，有空余时间再更新）

更新时间较长，建议找个空余时间更新,不更新也可以，但为避免后续安装其他东西出错最好更一下

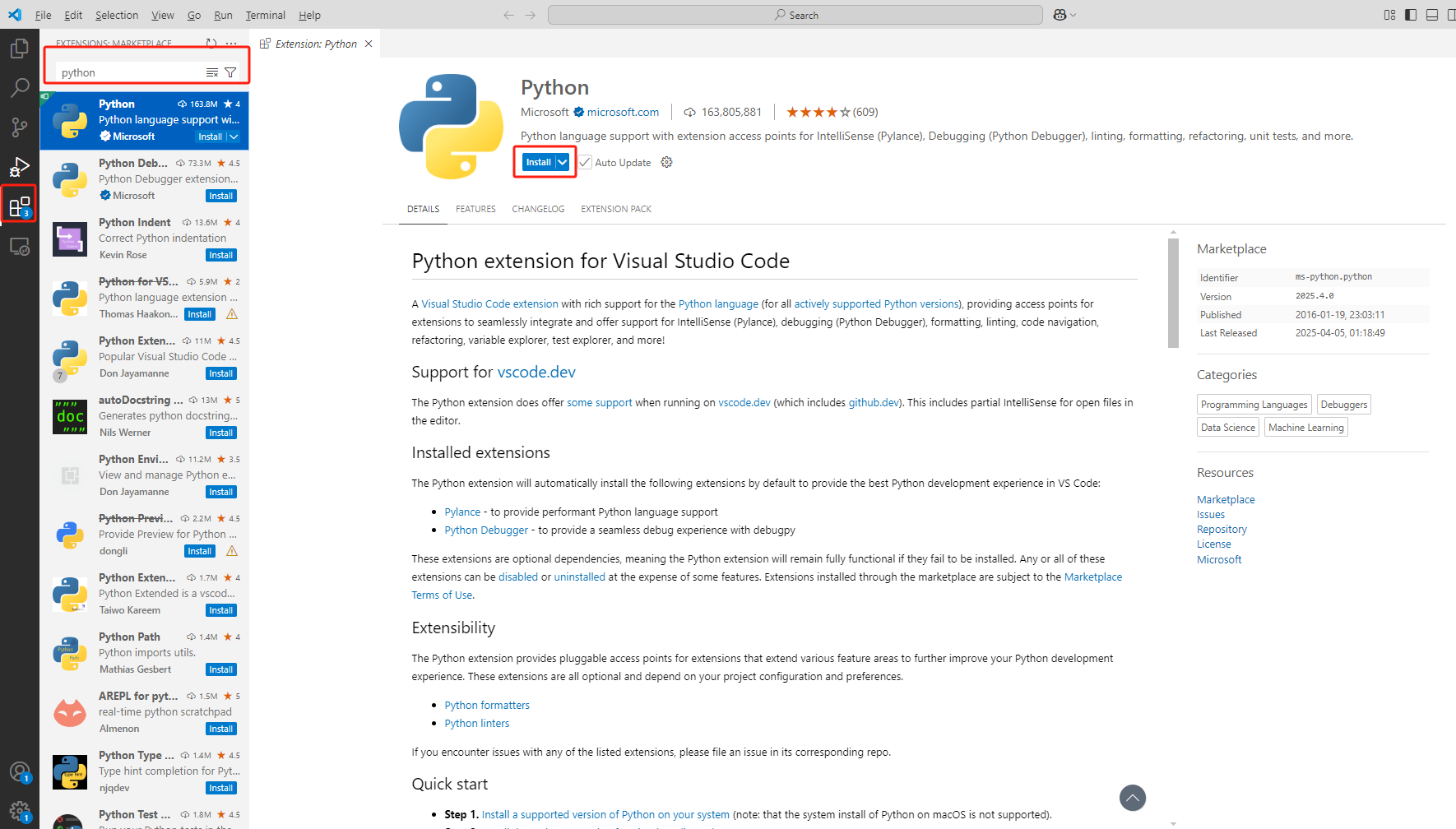
先更新conda

conda update conda

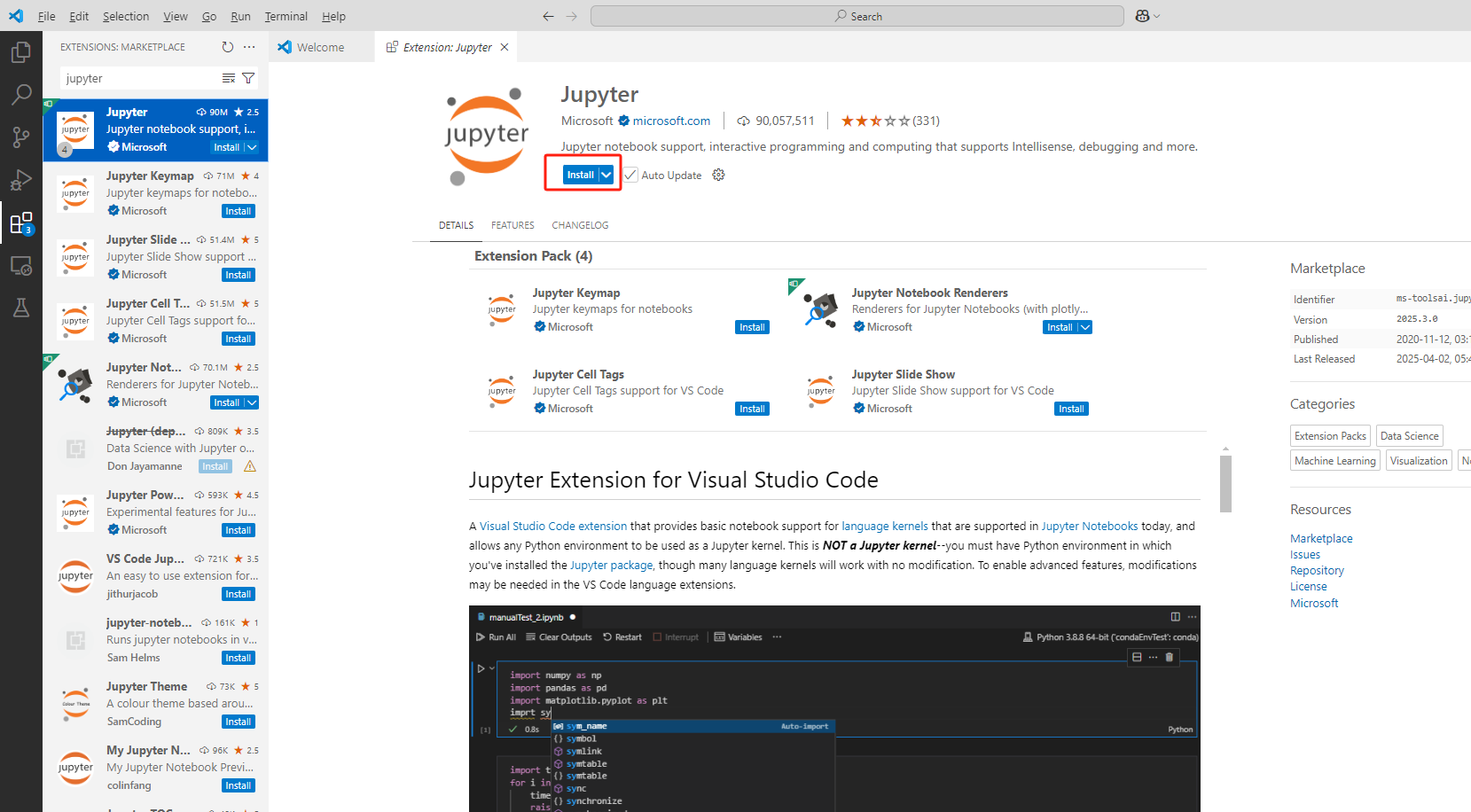


再更新第三方所有包

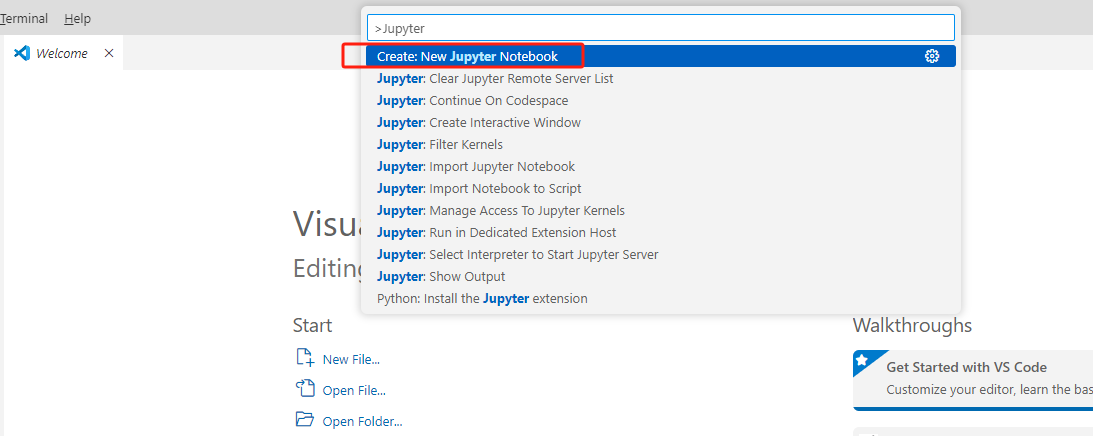
1. 配置VS Code
2. 安装Python插件



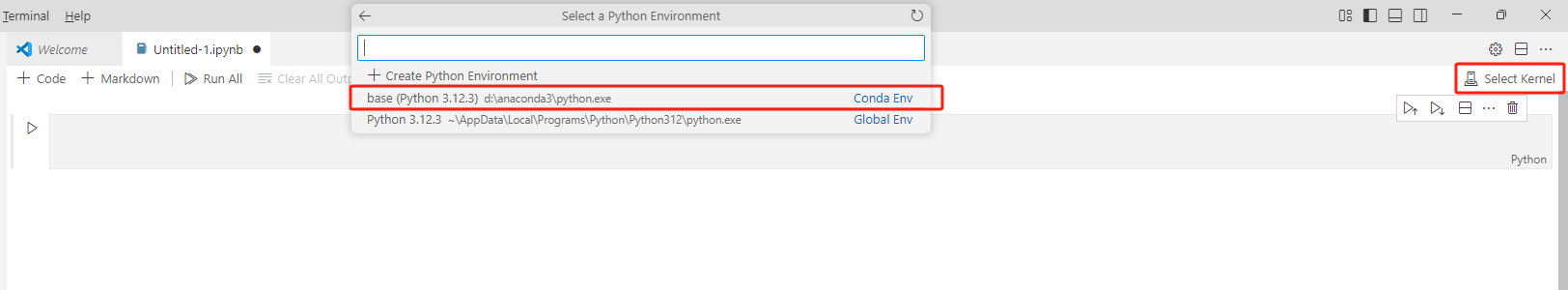
1. 安装Jupyter插件



1. 在VS Code中使用Jupyter Notebook
2. 新建Jupyter Notebook文件



1. 选择Kernel



1. 实验原理与步骤
2. 目的

知识图谱补全是从已知的知识图谱中提取出三元组(h,r,t)，为实体和关系进行建模，通过训练出的模型进行链接预测，以达成知识图谱补全的目标。

本文实验采用了FB15K-237数据集，分为训练集和测试集。利用训练集进行transE建模，通过训练为每个实体和关系建立起向量映射，并在测试集中计算MeanRank和Hit10指标进行结果检验。

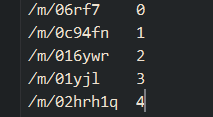
1. 数据集介绍

使用FB15K-237数据集

分为以下四个文件

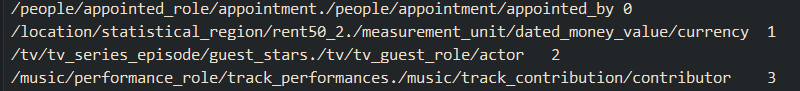
* **entity2id.txt**

实体和id对



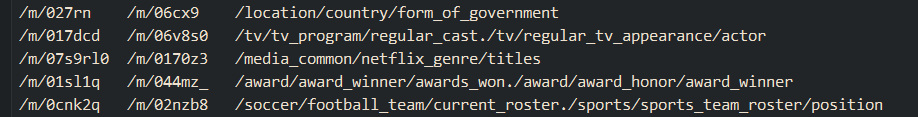
* **relation2id.txt**

关系和id对



* **train.txt**

训练集三元组（实体，实体，关系）



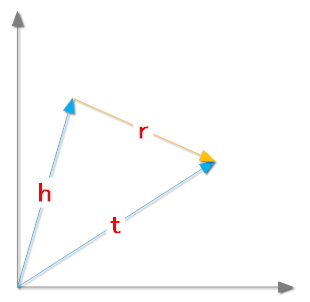
* **test.txt**

测试集三元组（实体，实体，关系）

1. 方法

TransE模型原理：

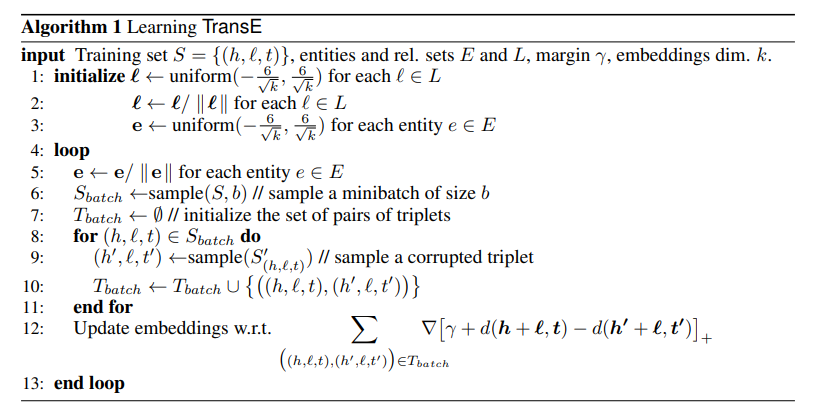
TransE将起始实体，关系，指向实体映射成同一空间的向量，如果（head,relation,tail）存在，那么h+r≈t



目标函数为：

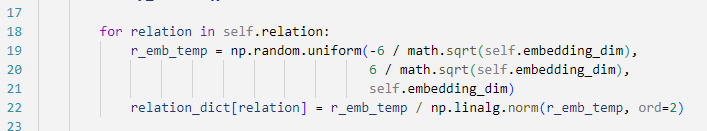


算法：



a) 初始化

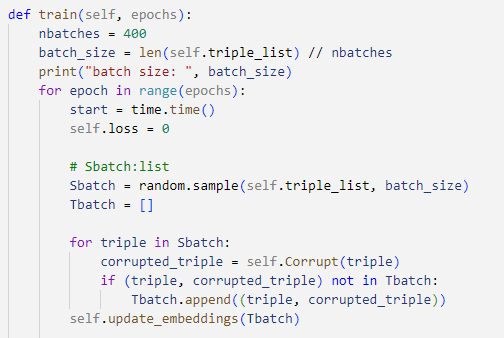
根据维度，为每个实体和关系初始化向量，并归一化



b) 选取batch

设置nbatches为batch数目，batch\_size = len(self.triple\_list) // nbatches

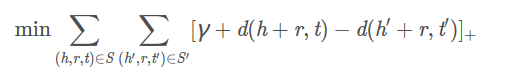
从训练集中随机选择batch\_size个三元组，并随机构成一个错误的三元组S'，进行更新



c) 梯度下降

定义距离\*d(x,y)\*来表示两个向量之间的距离，一般情况下，我们会取L1,或者L2 normal。

在这里，我们需要定义一个距离，对于正确的三元组\*(h,r,t),距离d(h+r,t)越小越好；对于错误的三元组(h',r,t'),距离d(h'+r,t')\*越大越好。

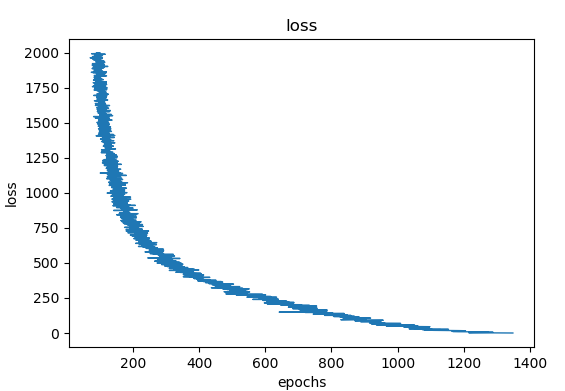


之后，使用梯度下降进行更新

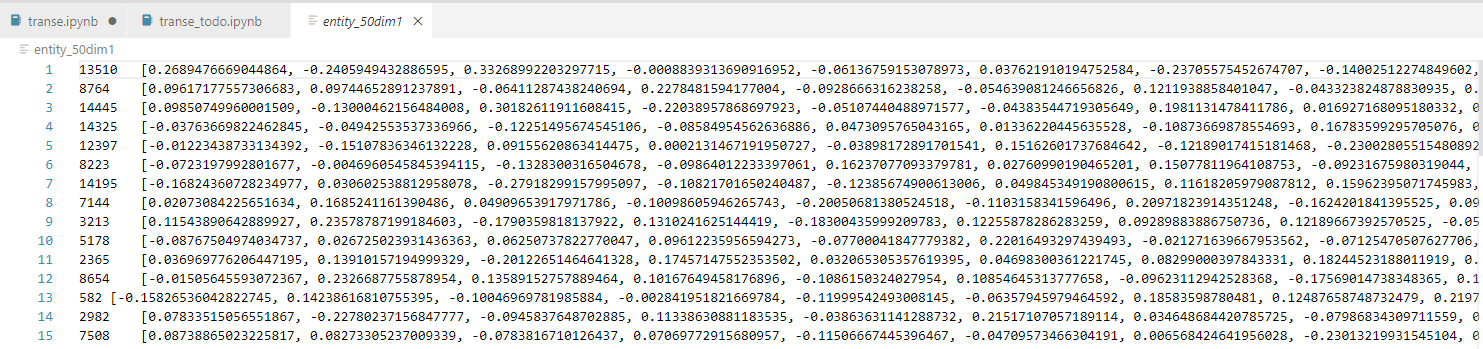
d) 结果

选择迭代次数2000次，向量维度50，学习率0.01进行训练

损失函数变化如下



结果存储在entity\_50dim1和relation\_50dim1中



e) 链接预测

通过transE建模后，我们得到了每个实体和关系的嵌入向量，利用嵌入向量，我们可以进行知识图谱的链接预测

将三元组(head,relation,tail)记为(h,r,t)

链接预测分为三类

1. **头实体预测：(?,r,t)**
2. **关系预测：(h,?,t)**
3. **尾实体预测：(h,r,?)**

但原理很简单，利用向量的可加性即可实现。以\*\*(h,r,?)\*\*的预测为例：

假设***t'=h+r***，则在所有的实体中选择与***t'距离最近的向量，即为t***的的预测值

f) 指标

Mean rank

对于测试集的每个三元组，以预测tail实体为例，我们将\*\*（h,r,t）\*\*中的t用知识图谱中的每个实体来代替，然后通过distance(h, r, t)函数来计算距离，这样我们可以得到一系列的距离，之后按照升序将这些分数排列。

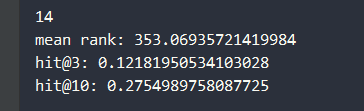
distance(h, r, t)函数值是越小越好，那么在上个排列中，排的越前越好。

现在重点来了，我们去看每个三元组中正确答案也就是真实的t到底能在上述序列中排多少位，比如说t1排100，t2排200，t3排60.......，之后对这些排名求平均，mean rank就得到了。

Hit@10

还是按照上述进行函数值排列，然后去看每个三元组正确答案是否排在序列的前十，如果在的话就计数+1

最终 排在前十的个数/总个数 就是Hit@10



g) 结果

经过TransE建模后，在测试集的13584个实体，961个关系的 59071个三元组中，测试结果如下：

mean rank: 353.06935721419984

hit@3: 0.12181950534103028

hit@10: 0.2754989758087725

一方面可以看出训练后的结果是有效的，但不是十分优秀，可能与TransE模型的局限性有关，TransE只能处理一对一的关系，不适合一对多/多对一关系。

1. 执行notebook代码，完成TODO任务

【实验任务】完成notebook文件transe\_todo.ipynb中的内容：

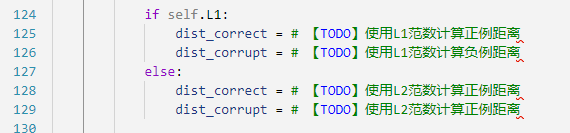
(1) 补全TODO代码

1. 初始化实体embedding



【补全后代码如下】：

1. 使用范数计算正负例距离



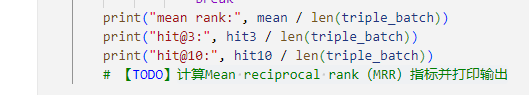
【补全后代码如下】：

c) 编写损失函数代码



【补全后代码如下】：

d) 计算Mean reciprocal rank（MRR）指标并打印输出



【补全后代码如下】：

(2) 将notebook每个代码段执行成功的输出截图粘贴在下面

【代码段1执行截图】：

【代码段2执行截图】：

……