Knowledge Embedding-OpenKE

- Conda安装及配置
- Pytorch安装
- OpenKE介绍及使用

前置实验环境

- Linux操作系统
  - 推荐稳定版本的Ubuntu
    - 如：Ubuntu 20.04.3 LTS
    - 下载链接： https://ubuntu.com/download/desktop
  - 可以在虚拟机中配置
    - VMWare:下载地址：
      https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html
    - VMWare激活码(选择任意一个使用即可)：
      ZF3R0-FHED2-M80TY-8QYGC-NPKYF
      YF390-0HF8P-M81RQ-2DXQE-M2UT6
      ZF71R-DMX85-08DQY-8YMNC-PPHV8

Conda简介

Conda是包及其依赖项和环境的管理工具

- 适用语言：Python, R, Ruby, Lua, Scala, Java, JavaScript, C/C++, FORTRAN
- 适用平台：Windows, macOS, Linux

用途:

- 快速安装、运行和升级包及其依赖项。
- 在计算机中便捷地创建、保存、加载和切换环境。

Conda安装

Anaconda下载安装

1.下载：$ wget https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/Anaconda3-2021.11-Linux-x86_64.sh

2.修改安装包权限：chmod 777 Anaconda3-2021.11-Linux-x86_64.sh

3.安装：$ ./Anaconda3-2021.11-Linux-x86_64.sh

```
(base) zjt@test-Super-Server:/data/zjt$ wget  https://repo.anaconda.com/archive/Anaconda3-2020.07-Linux-x86_64.sh
--2020-11-10 16:34:07--  https://repo.anaconda.com/archive/Anaconda3-2020.07-Linux-x86_64.sh
Resolving repo.anaconda.com (repo.anaconda.com)... 2606:4700::6810:8303, 2606:4700::6810:8203, 104.16.131.3, ...
Connecting to repo.anaconda.com (repo.anaconda.com)|2606:4700::6810:8303|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 576830621 (550M) [application/x-sh]
Saving to: 'Anaconda3-2020.07-Linux-x86_64.sh'

Anaconda3-2020.07-Linux-x86_64.sh                   15%[=================>
```

Conda安装

conda环境变量

安装结束会提示以下内容：

Do you wish the installer to initialize Anaconda3 by running

conda init?[yes|no]

1.输入yes，自动配置环境变量

conda换源

1.创建一个源文件

conda config --add channels r

2.使用vi进行编辑

vi .condarc

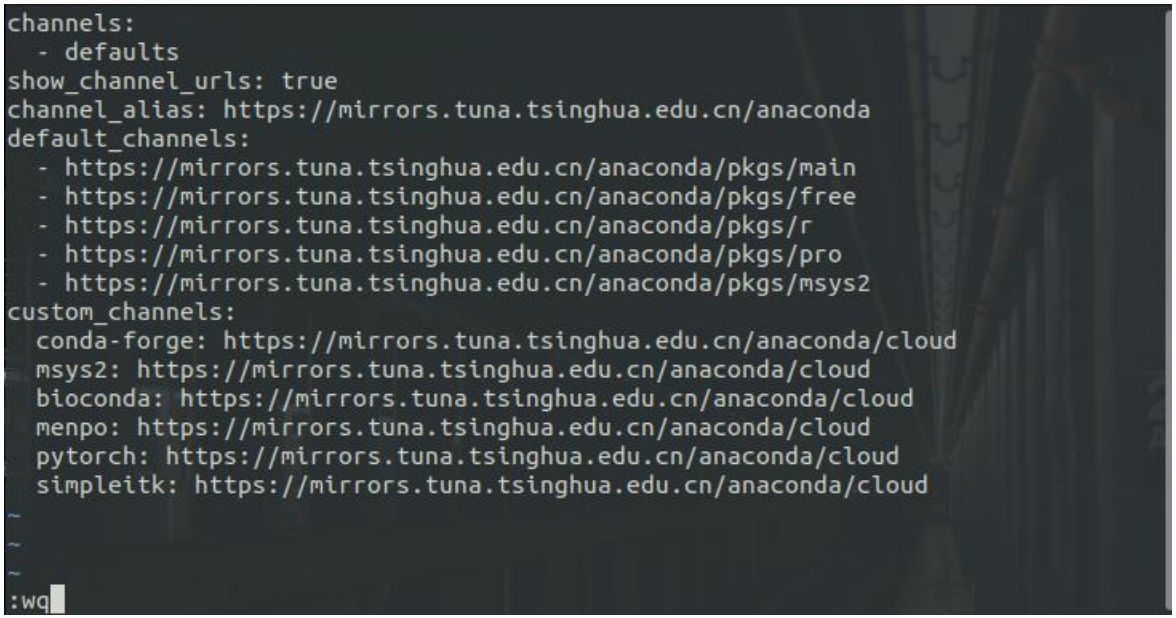
Conda安装

使用vi编辑.condarc文件

1.双击d键对内容进行删除，直至为空

2.粘贴source.txt文件内容

3.Esc退出编辑模式，输入：wq后按回车保存并退出



Conda安装

利用conda创建python环境

1. 创建python3.7环境： $ conda create -n kg python=3.7

2. 激活环境： $ conda activate kg

3.安装需要的tqdm和sklearn包

conda install tqdm

pip install sklearn

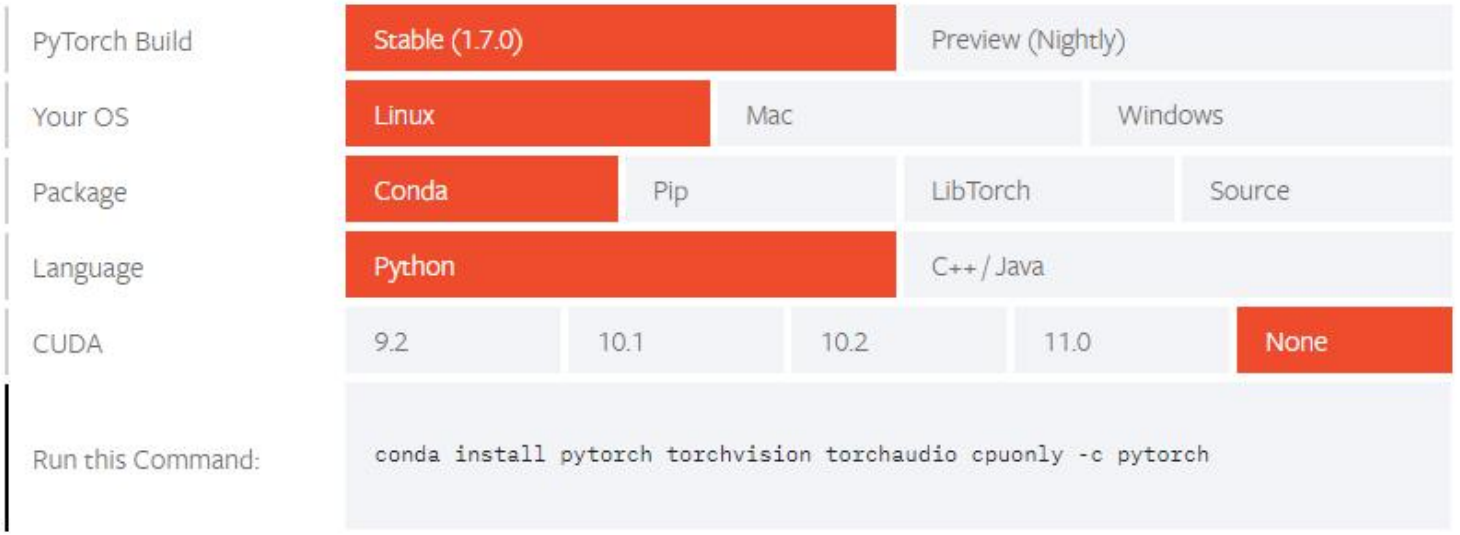
```
(base) zjt@test-Super-Server:~$ conda create -n kg python=3.7
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /data/zjt/anaconda3/envs/kg

  added / updated specs:
    - python=3.7


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    ca-certificates-2020.10.14 |                0         121 KB
    certifi-2020.6.20          |     pyhd3eb1b0_3         155 KB
    pip-20.2.4                 |   py37h06a4308_0         1.7 MB
    setuptools-50.3.1          |   py37h06a4308_1         711 KB
    ------------------------------------------------------------
                                           Total:         2.7 MB

The following NEW packages will be INSTALLED:

  _libgcc_mutex      pkgs/main/linux-64::_libgcc_mutex-0.1-main
  ca-certificates    pkgs/main/linux-64::ca-certificates-2020.10.14-0
  certifi            pkgs/main/noarch::certifi-2020.6.20-pyhd3eb1b0_3
  ld_impl_linux-64   pkgs/main/linux-64::ld_impl_linux-64-2.33.1-h53a641e_7
  libedit            pkgs/main/linux-64::libedit-3.1.20191231-h14c3975_1
  libffi             pkgs/main/linux-64::libffi-3.3-he6710b0_2
  libgcc-ng          pkgs/main/linux-64::libgcc-ng-9.1.0-hdf63c60_0
  libstdcxx-ng       pkgs/main/linux-64::libstdcxx-ng-9.1.0-hdf63c60_0
  ncurses            pkgs/main/linux-64::ncurses-6.2-he6710b0_1
  openssl            pkgs/main/linux-64::openssl-1.1.1h-h7b6447c_0
  pip                pkgs/main/linux-64::pip-20.2.4-py37h06a4308_0
  python             pkgs/main/linux-64::python-3.7.9-h7579374_0
  readline           pkgs/main/linux-64::readline-8.0-h7b6447c_0
  setuptools         pkgs/main/linux-64::setuptools-50.3.1-py37h06a4308_1
  sqlite             pkgs/main/linux-64::sqlite-3.33.0-h62c20be_0
  tk                 pkgs/main/linux-64::tk-8.6.10-hbc83047_0
  wheel              pkgs/main/noarch::wheel-0.35.1-py_0
  xz                 pkgs/main/linux-64::xz-5.2.5-h7b6447c_0
  zlib               pkgs/main/linux-64::zlib-1.2.11-h7b6447c_3


Proceed ([y]/n)? y


Downloading and Extracting Packages
pip-20.2.4          | 1.7 MB    | ##################################################################
ca-certificates-2020 | 121 KB    | ##################################################################
setuptools-50.3.1    | 711 KB    | ##################################################################
certifi-2020.6.20    | 155 KB    | ##################################################################
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

Pytorch简介

- PyTorch是一个基于Torch的Python开源机器学习库，用于自然语言处理等应用程序。它主要由Facebookd的人工智能小组开发，不仅能够实现强大的GPU加速，同时还支持动态神经网络，这一点是现在很多主流框架如TensorFlow都不支持的。PyTorch提供了两个高级功能：
  - 具有强大的GPU加速的张量计算（如Numpy）
  - 包含自动求导系统的深度神经网络

Pytorch安装

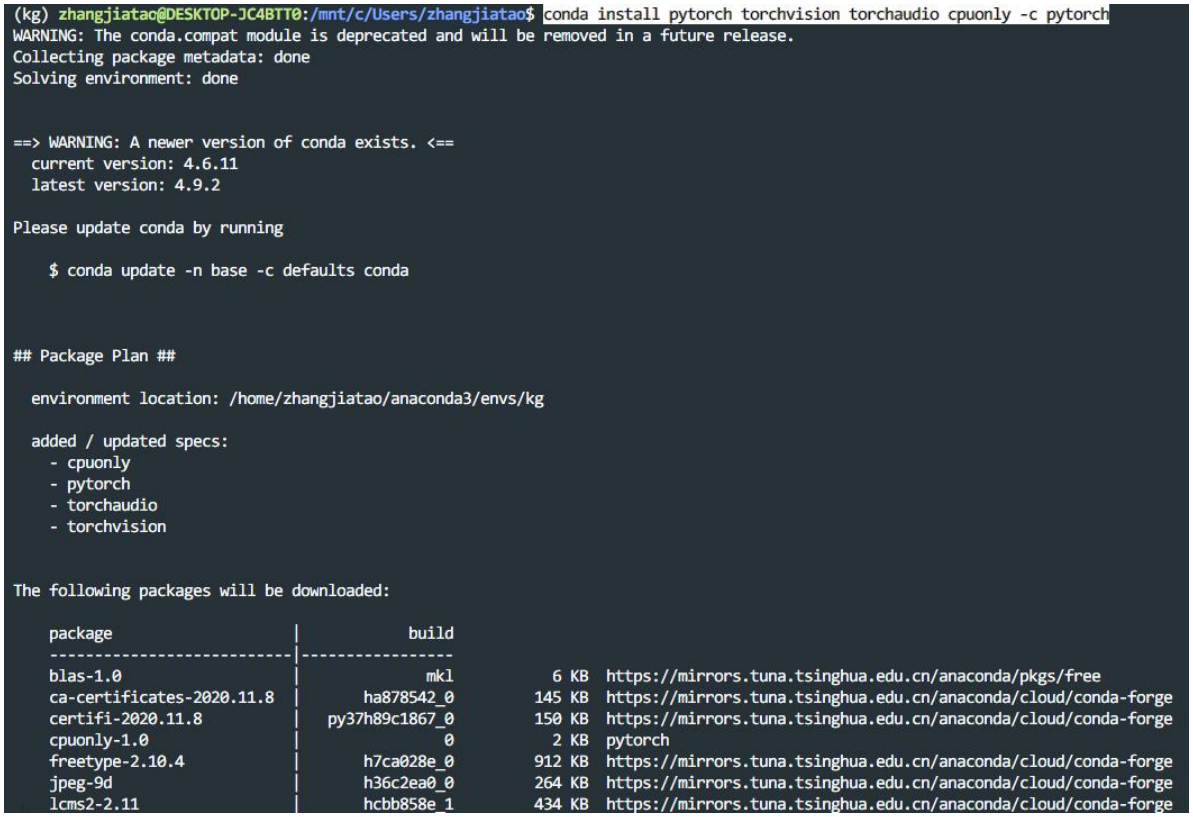
- 网址：https://pytorch.org/get-started/locally/
- 根据本机情况选择下载选项



Pytorch安装

输入生成的安装指令：

$ conda install pytorch torchvision torchaudio cpuonly -c pytorch



```
(kg) zhangjiatao@DESKTOP-JC4BTT0:/mnt/c/Users/zhangjiatao$ conda install pytorch torchvision torchaudio cpuonly -c pytorch
WARNING: The conda.compat module is deprecated and will be removed in a future release.
Collecting package metadata: done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.6.11
  latest version: 4.9.2

Please update conda by running

    $ conda update -n base -c defaults conda



## Package Plan ##

  environment location: /home/zhangjiatao/anaconda3/envs/kg

  added / updated specs:
    - cpuonly
    - pytorch
    - torchaudio
    - torchvision


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    blas-1.0                   |              mkl           6 KB  https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free
    ca-certificates-2020.11.8  |       ha878542_0         145 KB  https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge
    certifi-2020.11.8          |   py37h89c1867_0         150 KB  https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge
    cpuonly-1.0                |              0           2 KB  pytorch
    freetype-2.10.4            |       h7ca028e_0         912 KB  https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge
    jpeg-9d                    |       h36c2ea0_0         264 KB  https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge
    lcms2-2.11                 |       hcbb858e_1         434 KB  https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge
```

OpenKE简介

- OpenKE是THUNLP基于TensorFlow、PyTorch开发的用于将知识图谱嵌入到低维连续向量空间进行表示的开源框架。OpenKE提供了快速且稳定的各类接口，也实现了诸多经典的知识表示学习模型。该框架易于扩展，基于框架设计新的知识表示模型也十分的方便。

OpenKE配置

下载地址：

下载：https://github.com/thunlp/OpenKE

文档：http://139.129.163.161/static/index.html

运行样例：

```
Clone the OpenKE-PyTorch branch:
$ git clone -b OpenKE-PyTorch https://github.com/thunlp/OpenKE
$ cd OpenKE
$ cd openke
Compile C++ files
$ bash make.sh
Quick Start
$ cd ../
$ cp examples/train_transe_FB15K237.py ./
$ python train_transe_FB15K237.py
```

运行前的一些准备：

benchmarks同级目录下手动创建一个checkpoint文件夹来存放模型参数

修改py文件中的参数（根据配置选择性修改）

1.第39、45行改成：use_gpu=False

2.超参：nbatches=50 dim=100 train_times=10

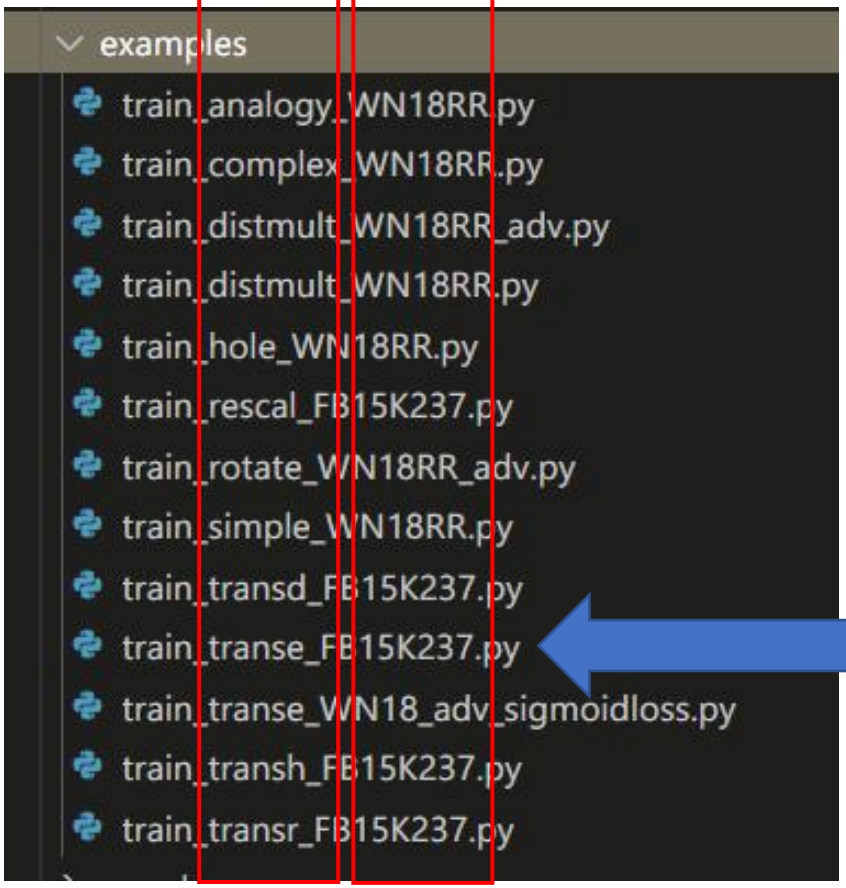
修改参数或给虚拟机多分配cpu都可以加快速度

- OpenKE简介
  - OpenKE结构
    - ./base/ .Cpp和.h文件，封装数据处理操作，向上层提供调用接口
    - Config.py 封装了训练、验证和预测相关的方法
      - 使用ctypes模块调用下层的C++数据处理接口
      - 使用torch中的optim、autograd模块统一进行模型反向传播和参数更新
    - Models.py作为所有实现模型的父类，继承自nn.Module，定义了四个通用方法
    - example_train_transe.py训练入口文件，最上层

示例-选择表示学习模型及数据集

examples目录下示例脚本

选择模型　　选择数据集



选择该脚本作为示例，
将该文件复制到
OpenKE根目录下

示例

示例脚本参数说明

```python
# dataloader for training
train_dataloader = TrainDataLoader(
    in_path = "./benchmarks/FB15K237/",
    nbatches = 100,
    threads = 8,
    sampling_mode = "normal",
    bern_flag = 1,
    filter_flag = 1,
    neg_ent = 25,
    neg_rel = 0)

# dataloader for test
test_dataloader = TestDataLoader("./benchmarks/FB15K237/", "link")
```

训练、测试数据参数

```python
# define the model
transe = TransE(
    ent_tot = train_dataloader.get_ent_tot(),
    rel_tot = train_dataloader.get_rel_tot(),
    dim = 200,
    p_norm = 1,
    norm_flag = True)


# define the loss function
model = NegativeSampling(
    model = transe,
    loss = MarginLoss(margin = 5.0),
    batch_size = train_dataloader.get_batch_size()
)
```

模型参数

```python
# train the model
trainer = Trainer(model = model, data_loader = train_dataloader, train_times = 1000, alpha = 1.0, use_gpu = True)
trainer.run()
transe.save_checkpoint('./checkpoint/transe.ckpt')

# test the model
transe.load_checkpoint('./checkpoint/transe.ckpt')
tester = Tester(model = transe, data_loader = test_dataloader, use_gpu = True)
tester.run_link_prediction(type_constrain = False)
```

训练参数

示例

Examples配置模型参数

```
# dataloader for training
train_dataloader = TrainDataLoader(
    in_path = "./benchmarks/FB15K237/",
    nbatches = 100,
    threads = 8,
    sampling_mode = "normal",
    bern_flag = 1,
    filter_flag = 1,
    neg_ent = 25,
    neg_rel = 0)

# dataloader for test
test_dataloader = TestDataLoader("./benchmarks/FB15K237/", "link")
```

- in_path: 训练集路径
- nbatches: Batch size
- threads: 线程数
- sampling_mode: 采样模型
- bern_flag: 负采样策略
- neg_ent: 打乱实体构造负例数量
- neg_rel: 打乱关系构造负例数量

示例

Examples配置模型参数

```
# define the model
transe = TransE(
    ent_tot = train_dataloader.get_ent_tot(),
    rel_tot = train_dataloader.get_rel_tot(),
    dim = 200,
    p_norm = 1,
    norm_flag = True)


# define the loss function
model = NegativeSampling(
    model = transe,
    loss = MarginLoss(margin = 5.0),
    batch_size = train_dataloader.get_batch_size()
)
```

- dim: Embedding size
- p_norm: 能量函数为1范数形式
- margin: Margin loss中margin值

示例

Examples配置模型参数

```
# train the model
trainer = Trainer(model = model, data_loader = train_dataloader, train_times = 1000, alpha = 1.0, use_gpu = True)
trainer.run()
transe.save_checkpoint('./checkpoint/transe.ckpt')

# test the model
transe.load_checkpoint('./checkpoint/transe.ckpt')
tester = Tester(model = transe, data_loader = test_dataloader, use_gpu = True)
tester.run_link_prediction(type_constrain = False)
```

- train_times: epoch-训练轮次
- alpha: 学习率
- use_gpu: 训练过程是否使用gpu加速

示例

运行脚本

```
(zjt_py) test-Super-Server% python train_transe_FB15K237.py
Input Files Path : ./benchmarks/FB15K237/
The toolkit is importing datasets.
The total of relations is 237.
The total of entities is 14541.
The total of train triples is 272115.
Input Files Path : ./benchmarks/FB15K237/
The total of test triples is 20466.
The total of valid triples is 17535.
Finish initializing...
Epoch 19 | loss: 5.382204:   2%|
⊗ 0 ⚠ 1   0
```

示例

输出测试结果

Hit@N：Hit@N 是用于排序任务（如搜索、推荐等）的评测指标。将知识图谱数据层事实性知识补全和知识图谱本体补全视为一个排序任务，即给定知识图谱三元组的任意两部分预测剩余的部分，例如给定三元组的主语和谓语预测三元组的宾语。

当预测出所有缺失部分时，按照相关性对预测结果进行排序并取前N个结果。假定所有需要预测的三元组个数为 K，而所有预测结果中正确结果在前N个结果的个数为K_j个，则计算Hit@N 如公式下所示：

$$Hits@N = \frac{K_j}{K}$$

```
no type constraint results:
metric:                 MRR         MR          hit@10      hit@3       hit@1
l(raw):                 0.261013    458.671143  0.557187    0.395924    0.073113
r(raw):                 0.261077    445.237366  0.554550    0.397278    0.072971
averaged(raw):          0.261045    451.954254  0.555868    0.396601    0.073042

l(filter):              0.402197    354.581635  0.801112    0.622960    0.133329
r(filter):              0.400600    334.621826  0.795553    0.616048    0.133115
averaged(filter):       0.401399    344.601746  0.798332    0.619504    0.133222
0.798332
0.7983324527740479
(zjt_py) test_Super_Server%
```

示例

输出测试结果

- MRR（Mean Reciprocal Rank）：首先计算每一个预测三元组中预测结果的位置排名得分 rankj如公式下所示：

$$MRR = \frac{1}{K}\sum_{j=1}^{K} rank_j$$

其中 R_j是预测正确结果的排名，然后针对所有 K个三元组计算 MRR 如公式如下所示：

$$rank_j = \frac{1}{R_j}$$

```
no type constraint results:
metric:                 MRR             MR              hit@10          hit@3           hit@1
l(raw):                 0.261013        458.671143      0.557187        0.395924        0.073113
r(raw):                 0.261077        445.237366      0.554550        0.397278        0.072971
averaged(raw):          0.261045        451.954254      0.555868        0.396601        0.073042

l(filter):              0.402197        354.581635      0.801112        0.622960        0.133329
r(filter):              0.400600        334.621826      0.795553        0.616048        0.133115
averaged(filter):       0.401399        344.601746      0.798332        0.619504        0.133222
0.798332
0.7983324527740479
(zjt_py) test_Super_Server%
```

示例

输出测试结果

- MR（Mean Rank）：与 MRR 类似，MR 也是评估整体排序质量的指标，MR 的计算公式如下所示：

  其中 R_j 是预测三元组正确结果的排名。

$$MR = \frac{1}{K}\sum_{j=1}^{K} R_j$$

```
no type constraint results:
metric:                 MRR             MR              hit@10          hit@3           hit@1
l(raw):                 0.261013        458.671143      0.557187        0.395924        0.073113
r(raw):                 0.261077        445.237366      0.554550        0.397278        0.072971
averaged(raw):          0.261045        451.954254      0.555868        0.396601        0.073042

l(filter):              0.402197        354.581635      0.801112        0.622960        0.133329
r(filter):              0.400600        334.621826      0.795553        0.616048        0.133115
averaged(filter):       0.401399        344.601746      0.798332        0.619504        0.133222
0.798332
0.7983324527740479
(zjt_py) test_Super_Server%
```

- 课堂作业
  - 阅读OpenKE README文档及样例程序，了解数据集结构及模型参数意义，训练满足以下条件的TransE、TransH模型，并获取测试性能，要求：
    - Embedding size = 100
    - 数据集为WN18RR
  - 其他参数可自行调整，
  - 将代码运行过程与实验结果(链接预测)截图形成实验报告。