**学习周报**

**2021.8.9-2021.8.15**

**汇报人：田昊**

1. 本周成果

运行TransE在数据集FB15K上的代码

了解传统模型

**The neural tensor network**——NTN

**KG embedding methods**——DistMult

**CNN models**——ConvE、ConvKB

继续学习KG-BERT代码

中文语料库

1. 本周总结

在中文语料库做知识补全的论文比较少，而且用于知识补全的中文数据集也不多

1. 下周计划

学习模型DKRL和ComplEx

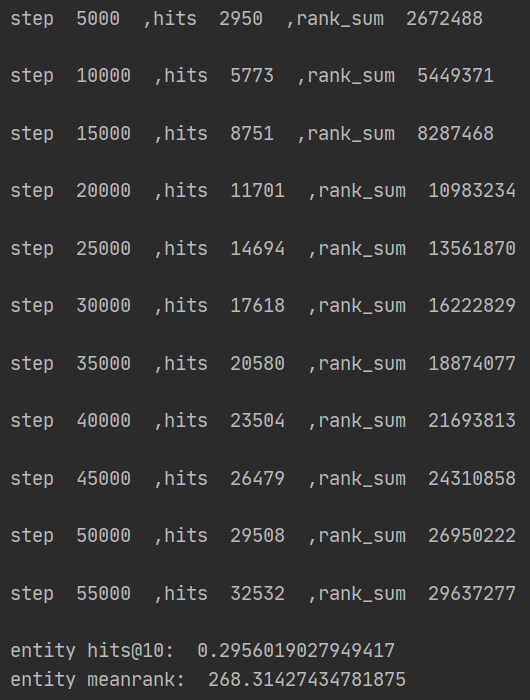
研究另两个任务的代码（大体和triple\_classification一样）

1. 学习笔记

运行TransE

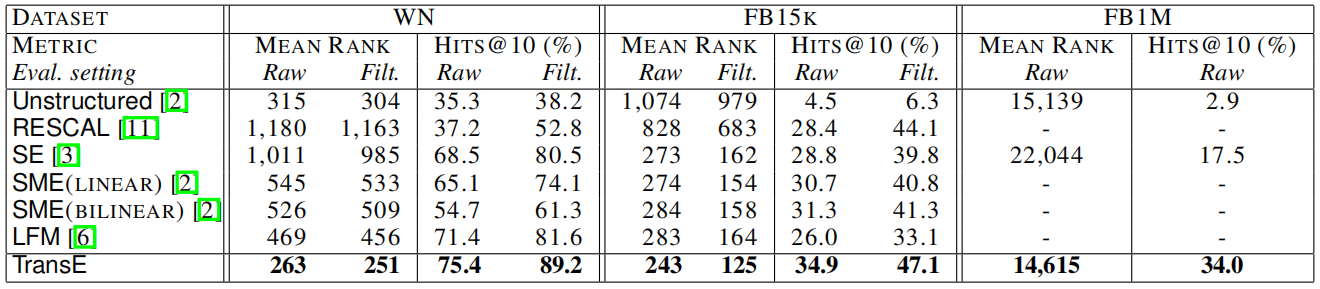
数据集：FB15k

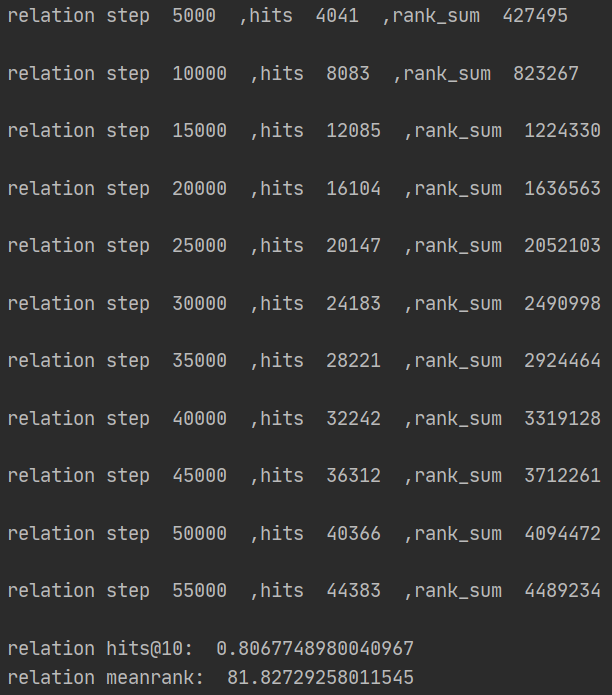
Link prediction任务



本地训练的结果MeanRank是268，Hit@10是0.296，与TransE原论文结果对比

TransE原论文结果是243和0.349

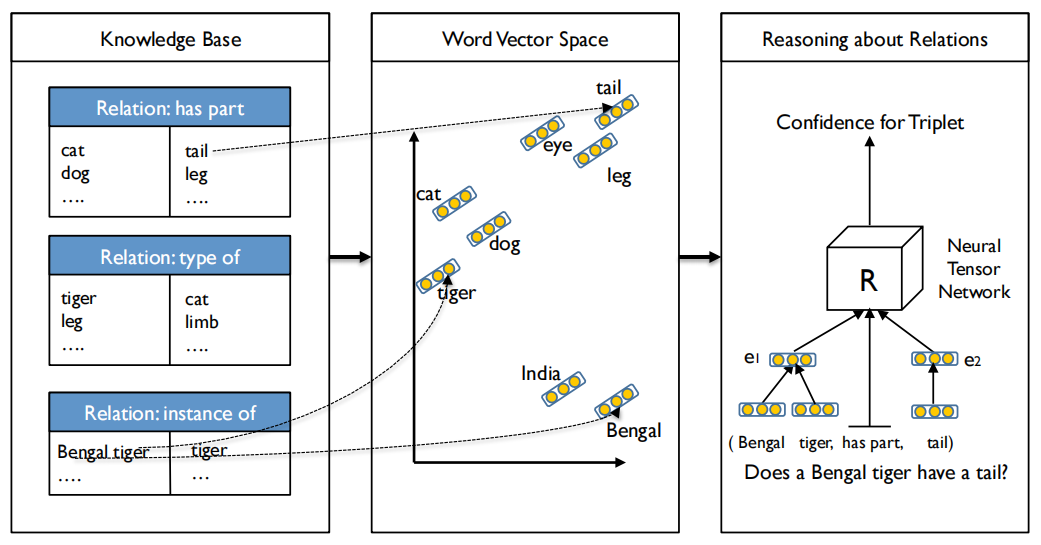




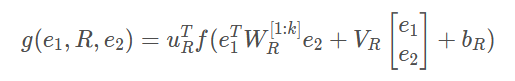
NTN模型——推理实体间关系

*Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. 2013. Reasoning with neural tensor networks for knowledge base completion. In NIPS, 926–934.*

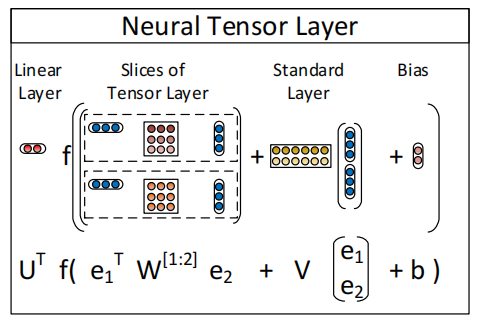
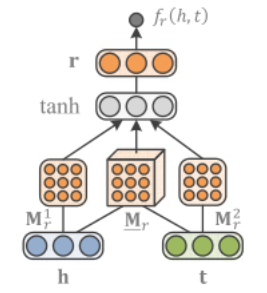
学习知识库中entity的向量表示（组成单词的平均值），以预测新的关系三元组



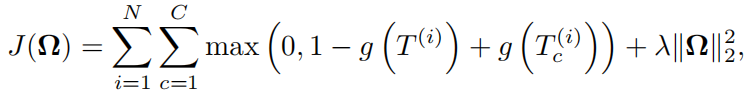
新关系三元组得分函数，u和V为权重，b为偏置，W为Tensor，f=tanh隐藏层激活函数：



偏置b和张量W由关系r特定，W每个切片负责一种实体对的实例化

损失函数，Ω为关系的NTN参数集Ω = u*,*W*,* V*,* b*,* E，N为训练集三元组数目，C为每个正确三元组生成的corrupted triple个数：



DistMult模型

*Bishan Yang, Wentau Yih, Xiaodong He, Jianfeng Gao,and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In ICLR*

张量分解（NTN）以及神经网络embedding（Trans）方法是表示学习的两种重要方法，这些方法的区别在于实体以及关系的表示方法

由于NTN的每个关系的需要参数较多，并且不能简单有效地处理大型的KGs

提出了一个一般化的多关系学习框架，涵盖了NTN、TransE等经典模型

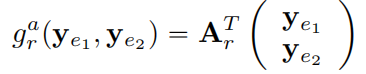
1. Entity representation



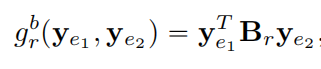
f是一个线性或者非线性的函数， W是一个随机初始化或者预训练的参数矩阵， ​和 ​是实体的向量one-hot表示，或者是实体中组成单词表示的平均值（NTN 模型）

1. Relation representation

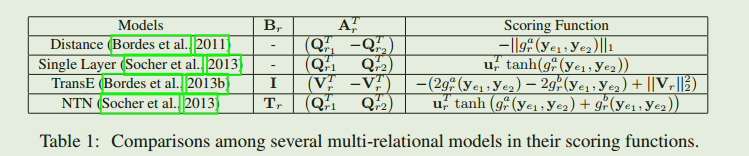
线性表示



双线性表示



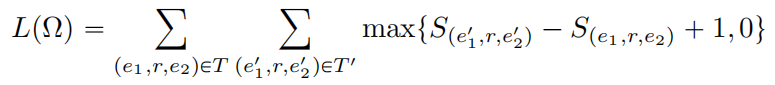
Ar和Br是特定的参数，以往方法的打分函数：



基于NTN模型，使用**对角矩阵**Mr简化得分函数：



损失函数：



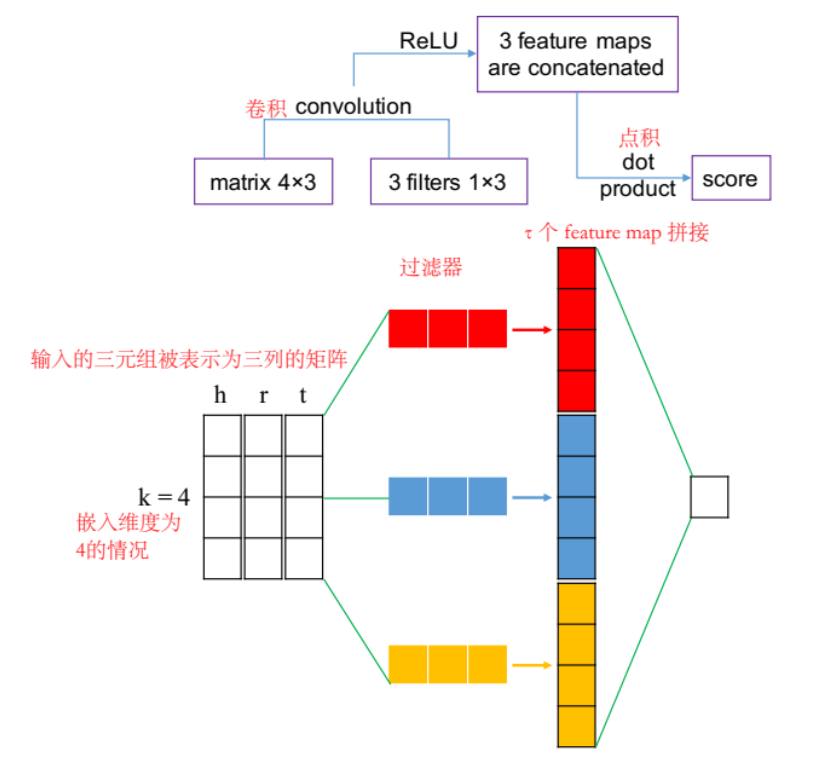
ConvKB 模型

*Nguyen, D. Q.; Nguyen, D. Q.; Nguyen, T. D.; and Phung, D. 2018. A convolutional neural network-based model for knowledge base completion and its application to search personalization. Semantic Web.*

*https://github.com/daiquocnguyen/ConvKB*

使用普通的一维卷积完成链接预测任务

每个三元组表示为一个三列的矩阵，输送到卷积层，多个卷积核对其进行卷积操作输出 feature maps，feature maps 拼接为一个单个的特征向量表示输入，特征向量与一个权重向量相乘，返回三元组得分。

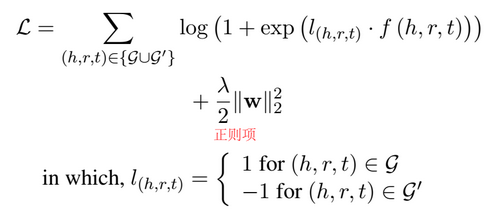


得分函数：



其中w和Ω是共享参数，\*代表卷积计算，g为激活函数

损失函数：



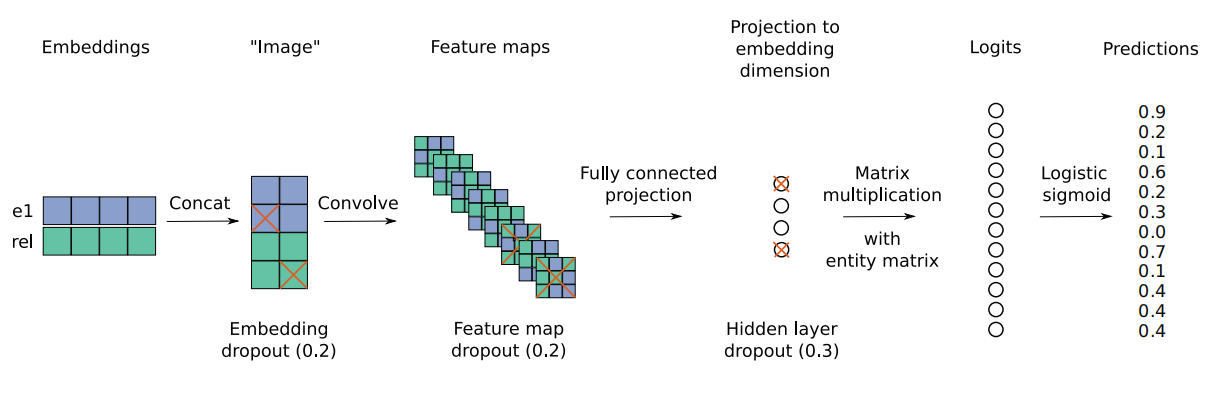
ConvE模型——第一个Convolutional Embedding

*Dettmers, T.; Minervini, P.; Stenetorp, P.; and Riedel, S. 2018. Convolutional 2d knowledge graph embeddings. In AAAI, 1811–1818.*

*https://github.com/TimDettmers/ConvE*

2D卷积操作来进行链接预测任务

在ConvE模型中，首先将头实体s和关系r的向量堆叠，然后 reshape 成二维的张量（类似图像的表示），经过2D卷积后，得到 feature map，然后经过全连接层投影到 k 维空间中，在内积层与候选目标的 embedding 进行匹配。

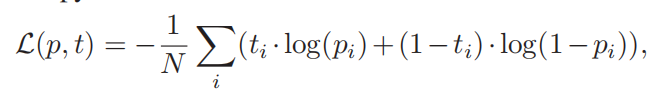


链接预测得分函数：



其中，是基于关系r的参数，是和的二维reshape的张量，矩阵W设置的线性变化将feature map投影到k维空间中。

二元交叉熵损失函数：



其中得分p是对ψ做sigmoid函数，定义如下：



KG-BERT-triple classification代码

|  |  |  |
| --- | --- | --- |
| 类、函数 | 参数 | 说明 |
| class InputExample | object | 数据预处理生成的字段名 |
| class InputFeatures | object | BERT输入格式 |
| class DataProcessor | object | 数据预处理 |
| class KGProcessor | DataProcessor | 知识图谱数据处理 |
| def convert\_examples\_to\_features | examples, label\_list, max\_seq\_length, tokenizer, print\_info=True | 将数据文件加载到InputBatch的列表中 |
| def \_truncate\_seq\_pair | tokens\_a, tokens\_b, max\_length | 截断序列对至最大长度 |
| def \_truncate\_seq\_triple | tokens\_a, tokens\_b, token\_c, max\_length | 截断三元序列对至最大长度 |
| def simple\_accuracy | preds, labels | 准确性 |
| def compute\_metrics | task\_name, preds, labels | 结果 |
| def main | / | 主函数 |

* 1. **class InputExample**

数据预处理后要生成的字段名

|  |  |
| --- | --- |
| guid | 示例的id |
| text\_a | untokenized text（head entity） |
| text\_b | 可选for sequence pair tasks（relation） |
| text\_c | 可选for sequence triple tasks（tail entity） |
| label | 标签，not for test examples |

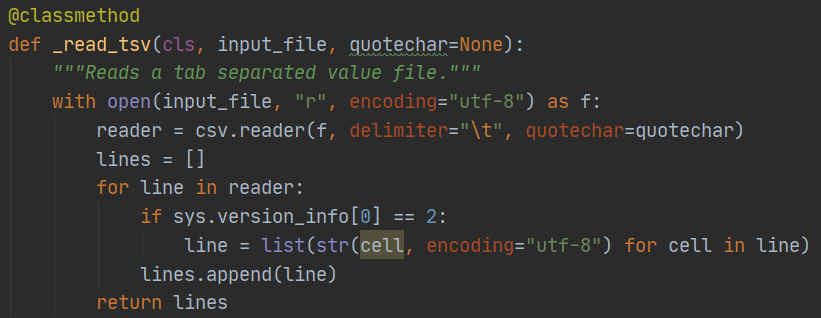
* 1. **class InputFeatures**

BERT基本输入字段

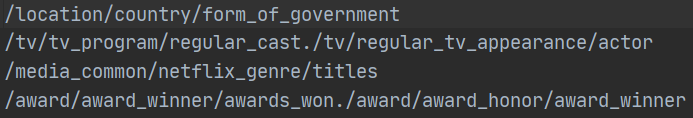
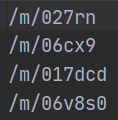
|  |  |
| --- | --- |
| input\_ids | 每个词id |
| input\_mask | real token为1，padding token为0 |
| segment\_ids | 句子级别的标签，区分h，r，t |
| label\_id | triple为1，corrupted triple为0 |

* 1. **class DataProcessor**

def \_read\_tsv 读取文件

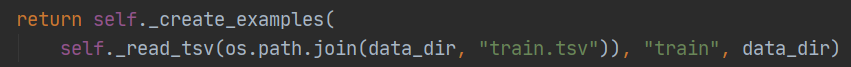


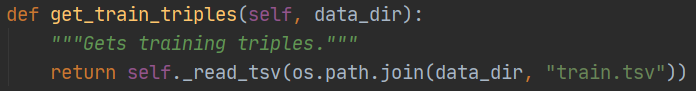
* 1. **class KGProcessor**
* get\_entities&relations&labels 获取KG中的entity，relation，标签（1,0）

* 获取examples和triples

get\_train\_examples&dev\_examples&test\_examples&train\_triples&dev\_triples&test\_triples





FB15K的triple（h，r，t）形式如下



* def \_create\_examples为训练和测试集创建示例

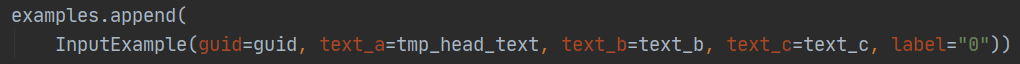
完成entity（relation）to text，读取entity（relation）2text文件





corrupting head and tail entity产生负样本





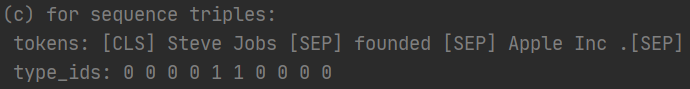
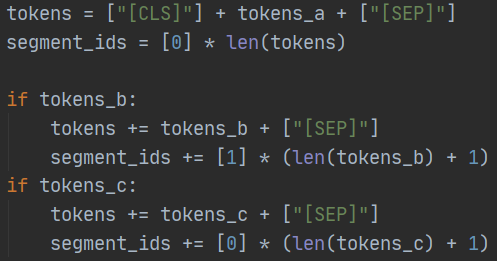
* 1. **def convert\_examples\_to\_features**

首先使用BertTokenizer.tokenize将example里的文本转换成它的单词段





Bert模型的输入形式，定义input\_ids，input\_mask，segment\_ids 和label\_ids

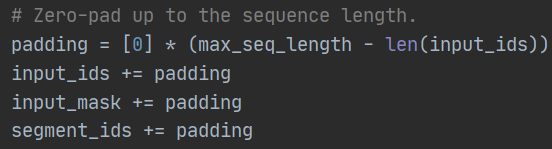




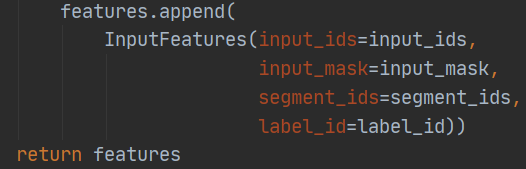




补0操作

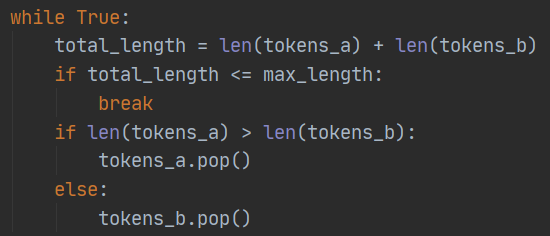


返回将4种字段



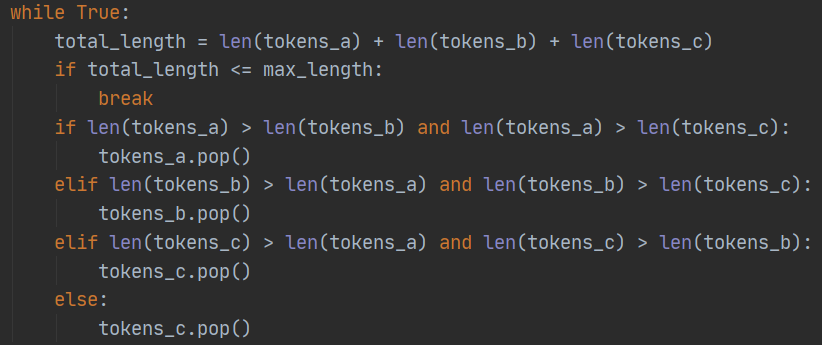
* 1. **def \_truncate\_seq\_pair**

每次截取长序列的一个元素

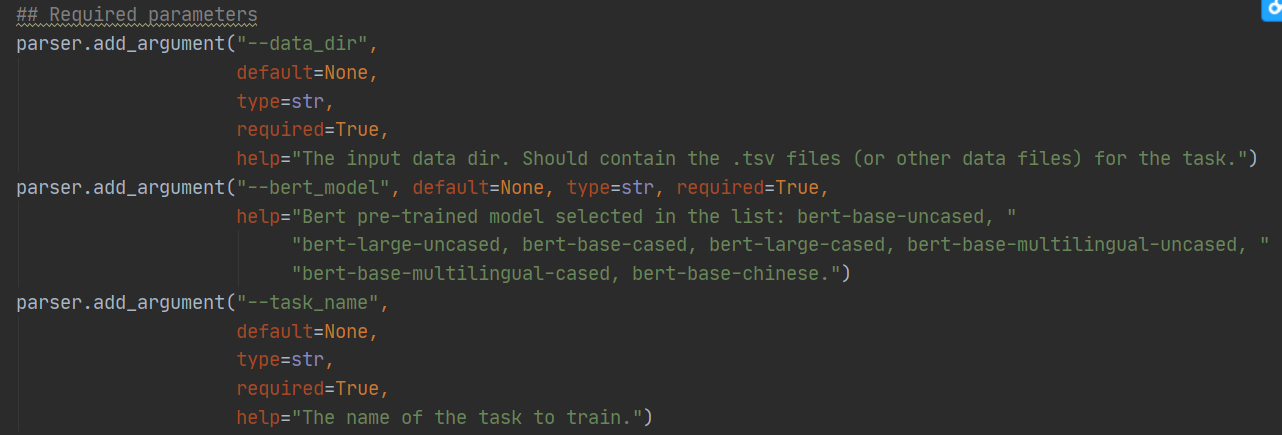


* 1. **def \_truncate\_seq\_triple**

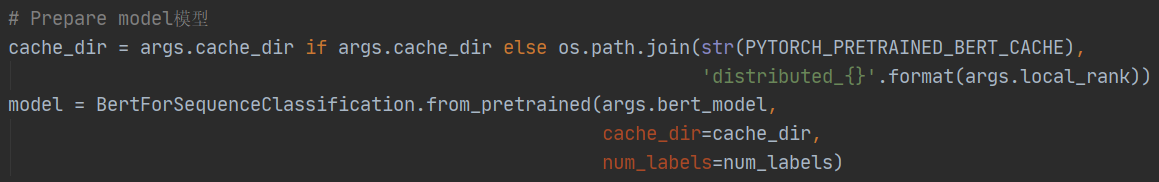
对于三元序列输入的截取



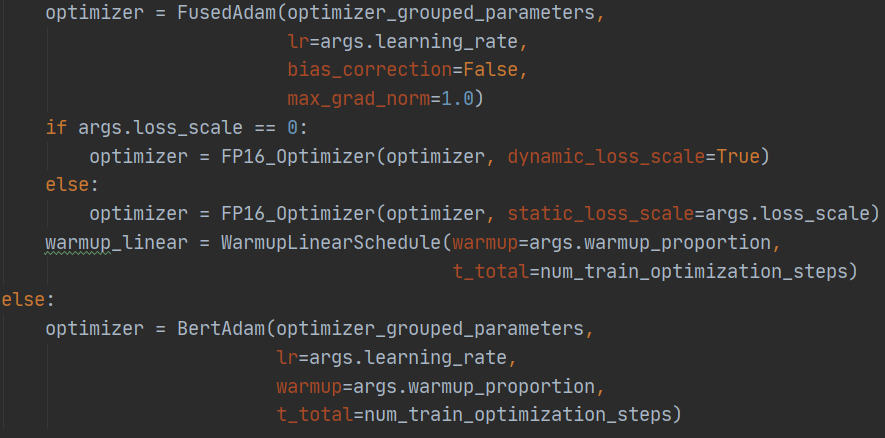
* 1. 最后的**main**函数分为4个部分
* 定义运行需要的参数



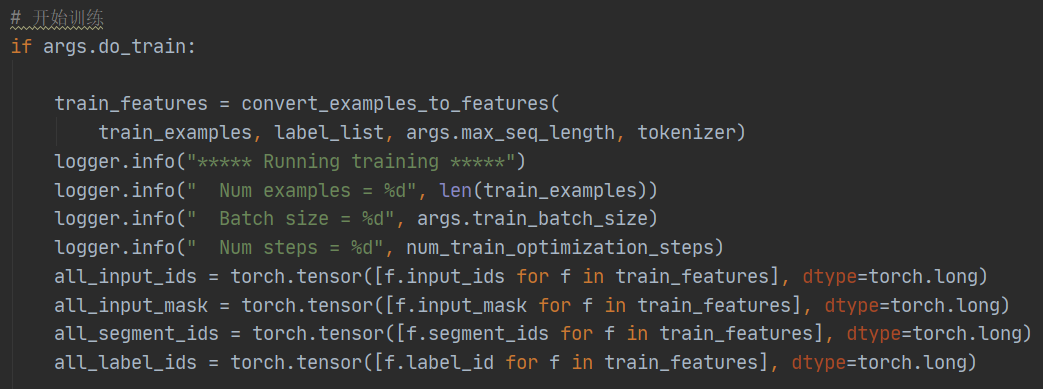
* 准备bert模型



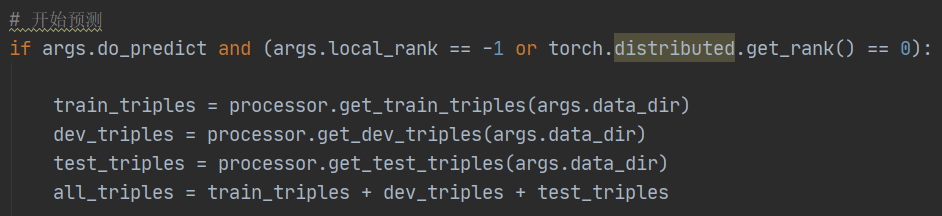
* 准备优化器



* 训练、验证、测试







中文语料库

* ChineseSemanticKB



* 中文百科知识图谱Zhishi.me

融合了三大中文百科，百度百科，互动百科以及维基百科中的数据

* 网站OpenKG



词向量长度：768