人工智能与数据挖掘课程设计

基于深度学习Bert模型的命名实体识别

**组员信息：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 姓名 | 学号 | 成员贡献 | 实践成绩 |
| 1 | 刘思蕊 | 2021211918 | 1/3 |  |
| 2 | 李雨霏 | 2021211919 | 1/3 |  |
| 3 | 丁宁 | 2021211924 | 1/3 |  |

# **引言**

（1）命名实体识别任务分析

命名实体识别（NamedEntityRecognition，简称NER），也称为专名识别，是自然语言处理（NLP）领域的一项关键技术。它的主要任务是在文本数据中识别出具有特定意义的实体，这些实体通常包括人名、地名、机构名、专有名词等。NER的目的在于将这些具有特定意义的实体提取出来，以便于后续的知识挖掘、信息抽取等任务。

命名实体识别的过程主要包括两个部分：

实体边界识别：确定文本中实体的起始和结束位置。例如，在句子“奥巴马是美国总统”中，需要识别出“奥巴马”和“美国”这两个实体的边界。

实体类别确定：判断识别出的实体属于哪个类别，如人名、地名、机构名等。在上述例子中，奥巴马和美国分别属于人名和地名类别。

目前，NER技术主要分为基于规则和字典的方法、基于统计模型的方法以及基于深度学习的方法。

基于规则和字典的方法：通过预先制定一套规则或构建一个实体词典，对文本进行扫描和匹配，从而实现实体的识别。这种方法的优点是易于实现，但缺点是扩展性差，对于新出现的实体或复杂场景难以应对。

基于统计模型的方法：通过对大量已标注的文本数据进行学习，训练出一种统计模型，用于对新的文本进行实体识别。常见的统计模型包括条件随机场（CRF）、支持向量机（SVM）等。这种方法的优点是具有一定的泛化能力，但缺点是需要大量的人工标注数据和特征工程。

基于深度学习的方法：利用深度学习技术，如卷积神经网络（CNN）、循环神经网络（RNN）、长短时记忆网络（LSTM）等，对文本进行端到端的建模，实现实体的识别。这种方法的优点是自动提取特征，减少人工干预，但缺点是需要大量的训练数据和计算资源。

（2）命名实体识别方法

a.BERT模型。BERT模型是基于TransformersEncoder，使用MLM获取双向融合信息，在海量连续语料上进行预训练，得到的预训练模型，基于下游任务只需简单的fine-tuning就能获得特别好的结果。

b.LSTM（循环神经网络）算法。是一种特殊的递归神经网络。这种网络与一般的前馈神经网络不同，LSTM可以利用时间序列对输入进行分析。对于LSTM，对于序列中的每个元素，有相应的隐藏状态ht，原则上可以包含序列中较早的任意点的信息。

c.CRF（条件随机场）算法。CRF（条件随机场）算法是给定一组输入序列条件下另一组输出序列的条件概率分布模型。CRF中有两类特征函数，分别是状态特征和转移特征。前者只针对当前位置的字符可以被转换成哪些实体label，后者关注的是当前位置和其相邻位置的字符可以有哪些实体label的组合。

# **实验数据**

## 数据获取或预处理

（1）数据获取

数据来源于人民日报NER数据集

简介：本NER数据集由人民日报语料库1998版和2014版生成，包含了人名(PER)、地名(LOC)和机构名(ORG)3类常见的实体类型。

语种：Chinese

实体类别数量：3

链接：

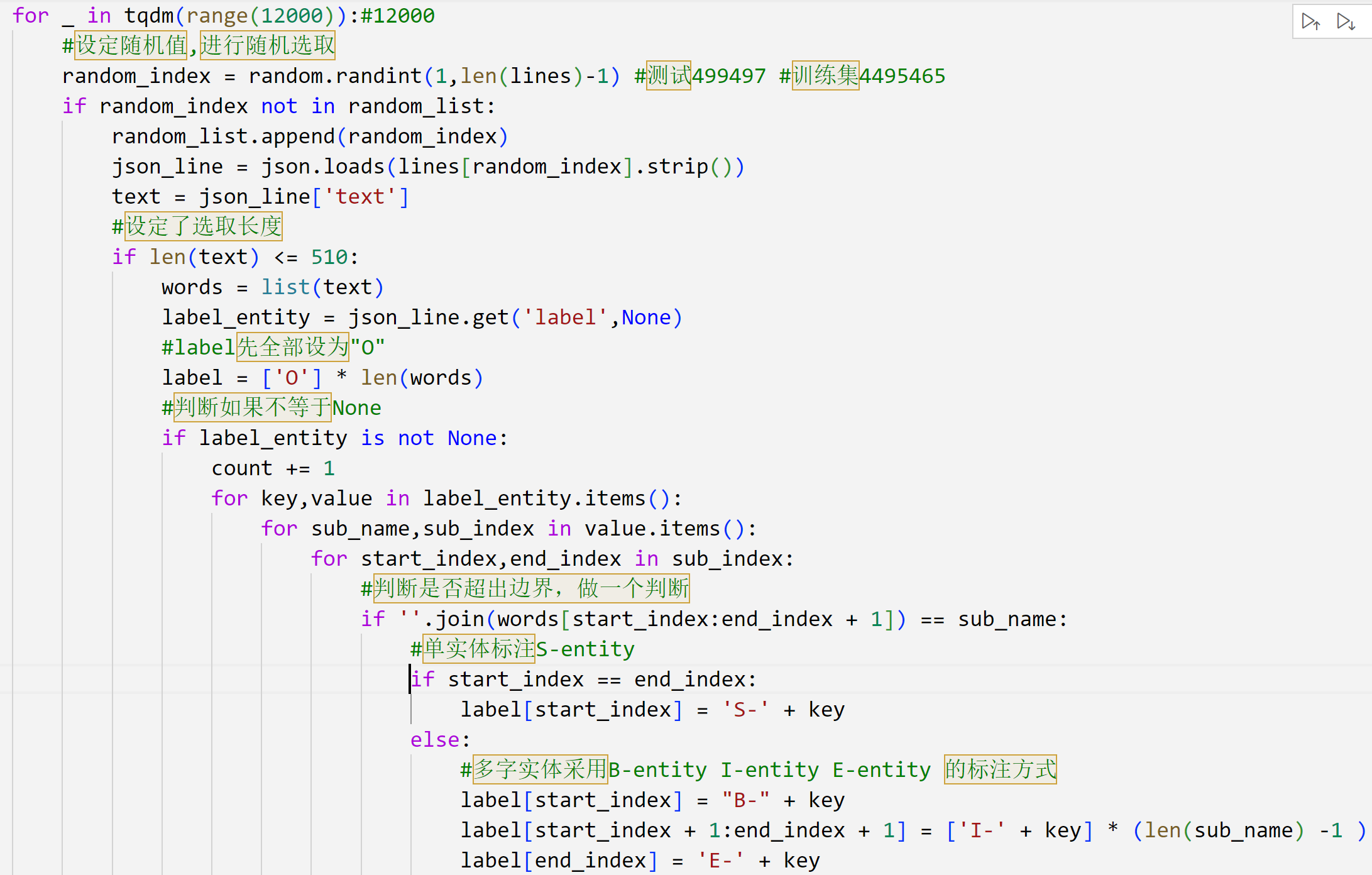
<https://github.com/InsaneLife/ChineseNLPCorpus/tree/master/NER/renMinRiBao>

<https://raw.githubusercontent.com/InsaneLife/ChineseNLPCorpus/master/NER/renMinRiBao/renmin.txt>

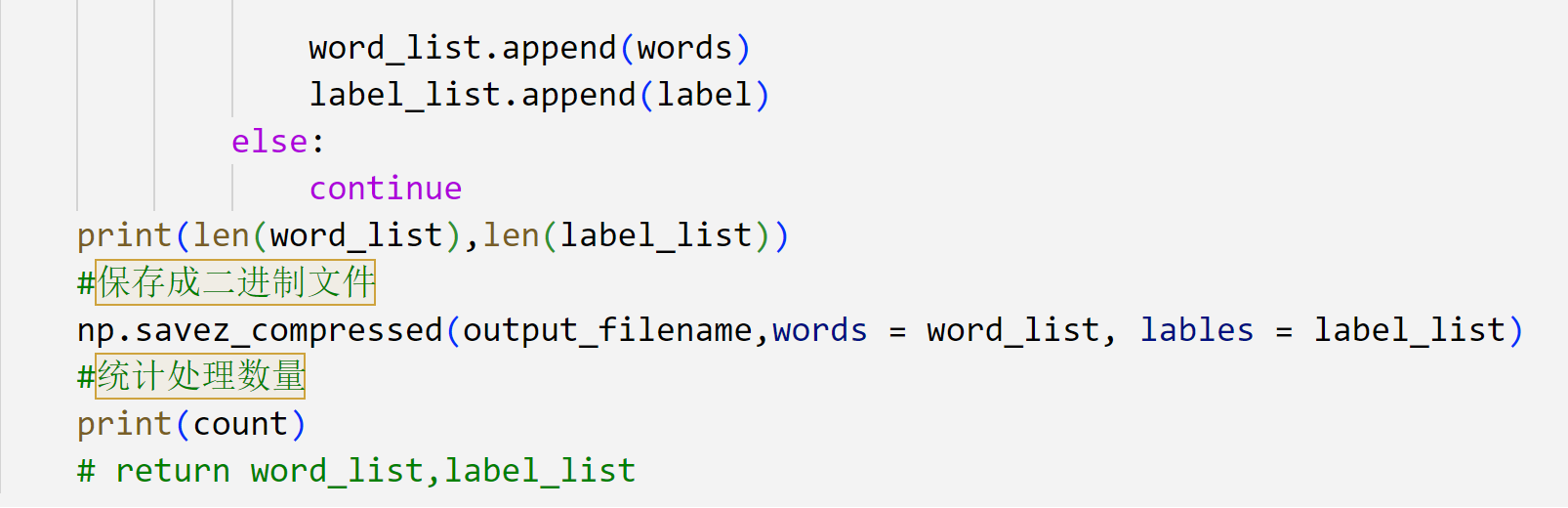


（2）数据预处理

a.单实体标注S-entity，多字实体采用B-entityI-entityE-entity的标注方式。具体代码如下：

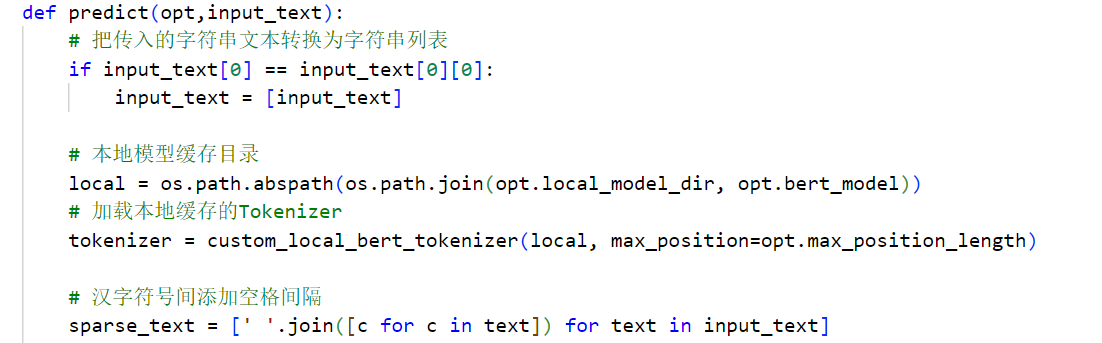


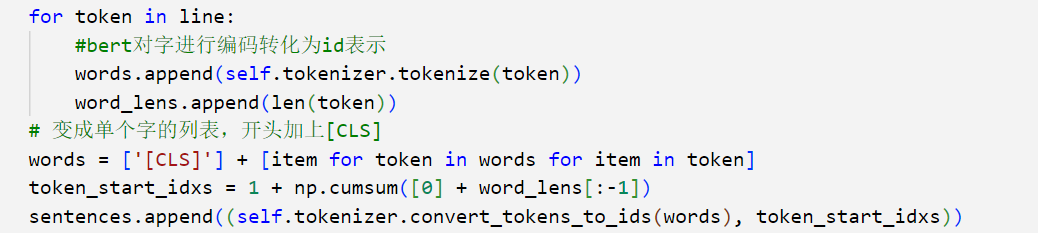
b.保存为二进制文件并统计数据处理数量。

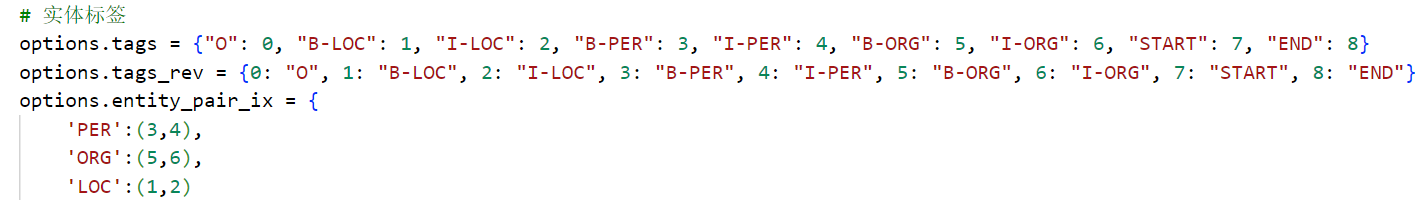


## 数据统计

（1）首先，需要将文本数据转换为模型可以处理的格式。BERT模型的输入是tokenizer的文本，即将文本分割成一个个单词或子词，并为每个单词或子词分配一个唯一的ID。同时，还需要为每个词标注上相应的实体标签。部分代码和结果如下：





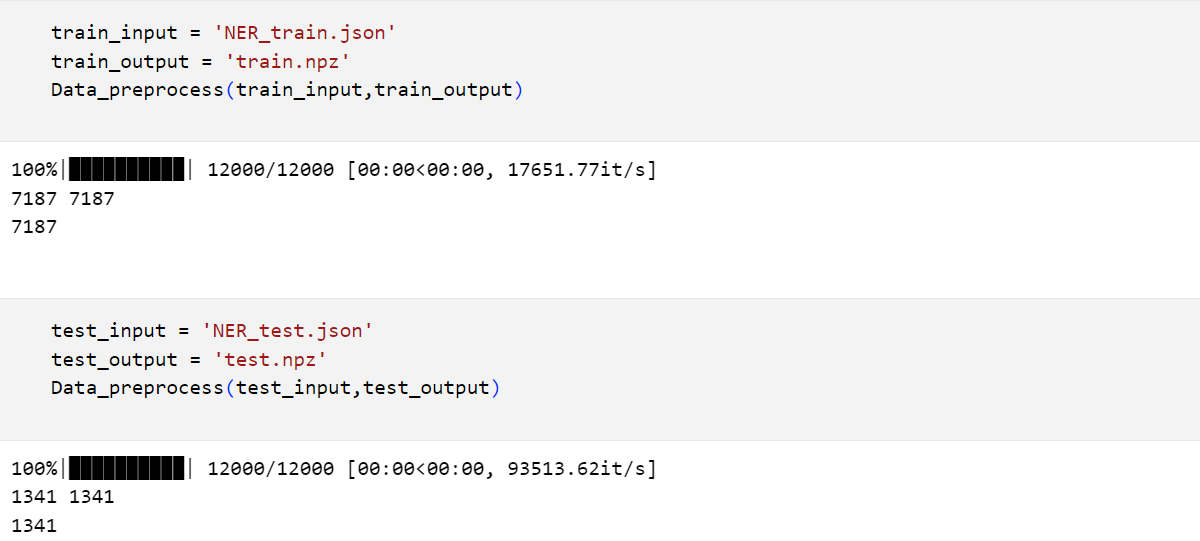




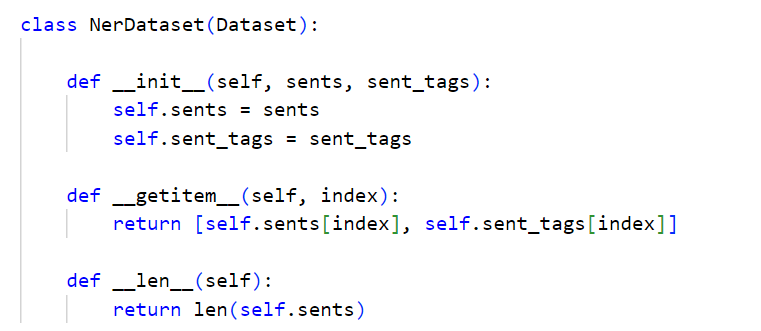


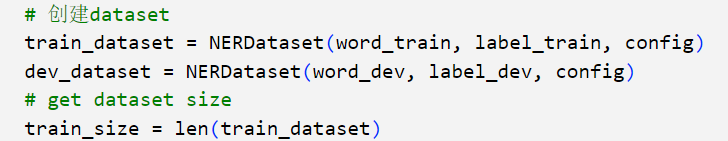
（2）接下来，需要将数据划分为训练集、验证集和测试集。可以按照一定的比例将数据集划分为这三部分，并将它们保存为不同的文件。



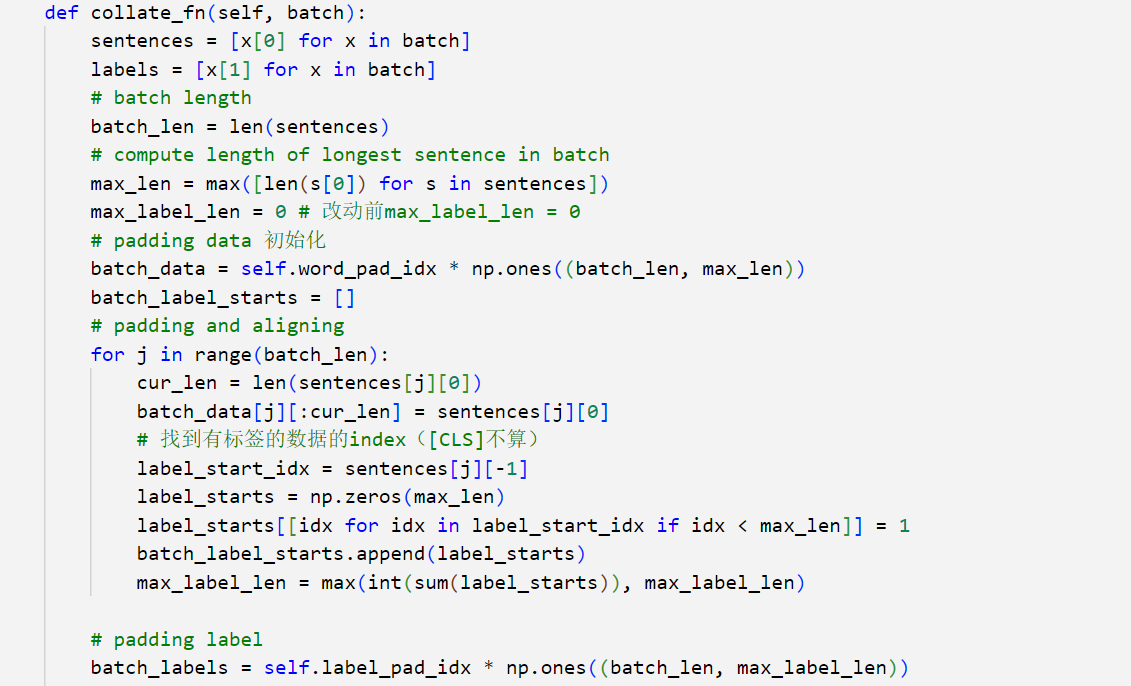


（3）在PyTorch中，可以使用Dataset和DataLoader来加载数据。首先，需要定义一个自定义的Dataset类，其中需要实现\_\_len\_\_和\_\_getitem\_\_方法。\_\_len\_\_方法返回数据集的大小，\_\_getitem\_\_方法返回指定索引的数据样本。在\_\_getitem\_\_方法中，可以根据索引读取对应的文本和标签数据，并进行相应的处理。具体代码如下：

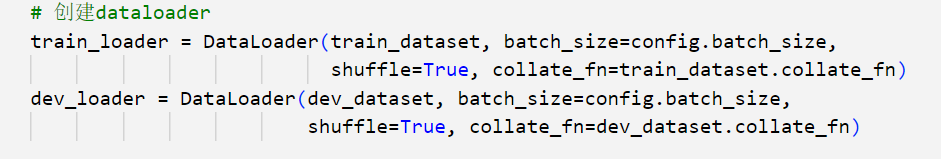




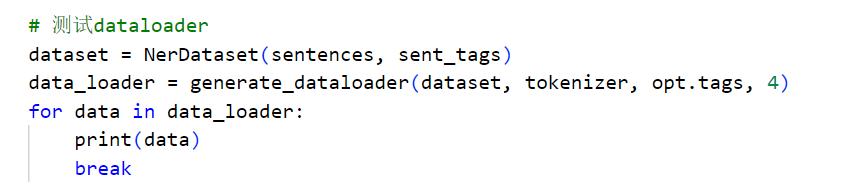
（4）在Dataset类中，还可以定义一些其他的数据处理方法，例如对文本进行截断、填充等操作，以适应模型的输入要求。例如，在序列长度不足的情况下可以填充0，或者使用截断方法保持一致的序列长度。



（5）定义完Dataset类后，可以使用DataLoader来加载数据。DataLoader可以指定批量大小、是否打乱数据等参数，方便进行训练和验证。







（6）最后，进行模型训练和预测，同时可以根据评价指标（如准确率、召回率和F1值）来评估模型的性能，并根据需要进行调整和优化。（详细描述在下部分内容）

# **方法/模型**

（1）模型设计

使用Bert作为底层的特征提取器，并加入双向lstm与线性层进行分类获得每个标签的预测类别，最后将其送入到crf中，根据发射分数和状态转移矩阵获得最佳的标签类别。

（2）模型训练流程

a.模块导入和设置

首先导入模型训练所需要的库和模块，例如Pytorch、transformers、torchmetrics以及一些自定义的模块和函数。创建TensorBoard的SummaryWriter，并且注意忽略UserWarning。

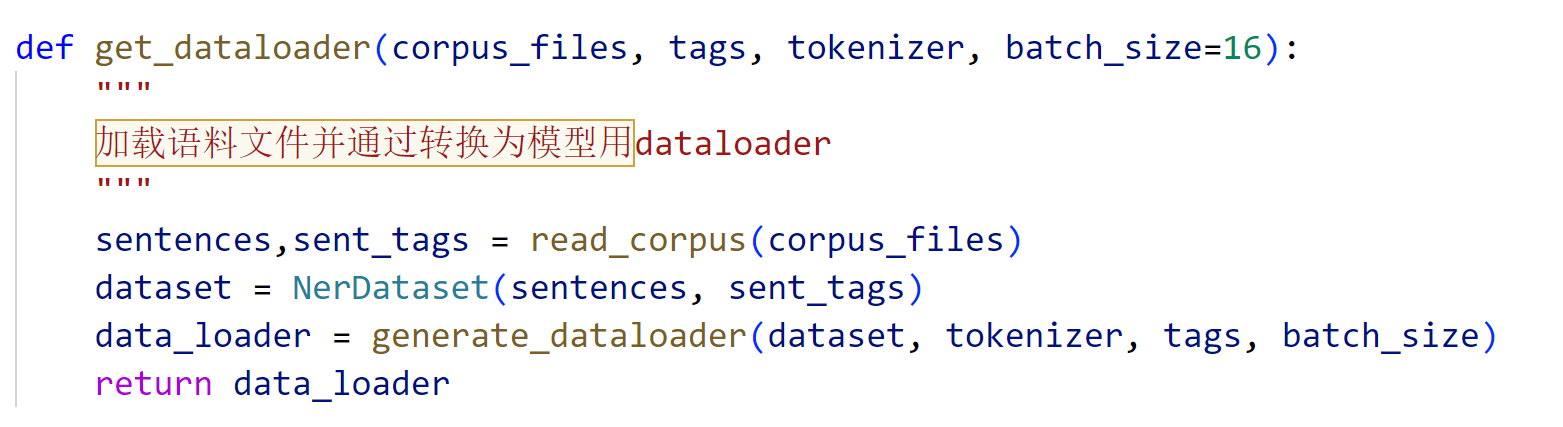
该部分主要导入了自定义的BertBiLstmCRF模型，该模型结合了BERT、双向LSTM和CRF（条件随机场）层，用于NER任务。同时使用TorchMetrics库进行性能指标的计算，使用TensorBoard进行训练过程的可视化。



b.数据加载和处理

通过定义函数读取语料文件，生成模型需要的dataloader。

get\_dataloader函数用于加载语料文件并生成模型所需的DataLoader，其中调用了自定义的read\_corpus函数和generate\_dataloader函数。



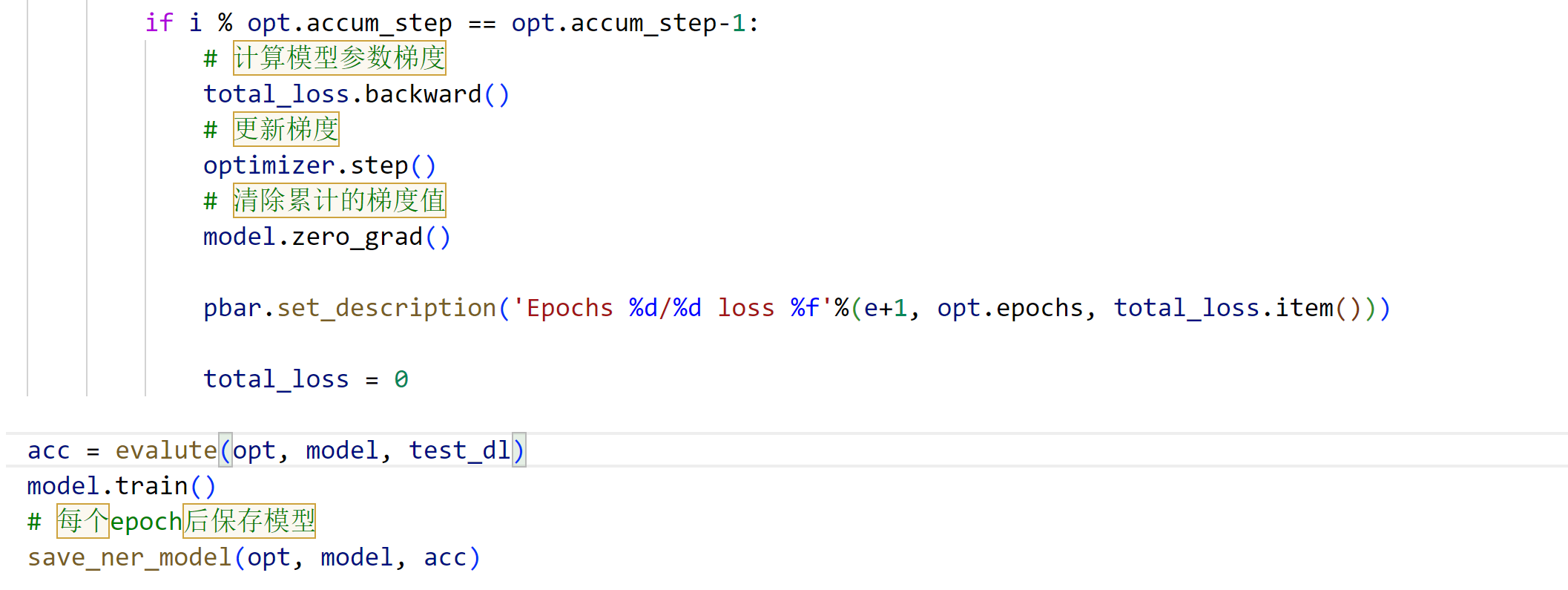
c.定义模型训练函数

设置优化器AdamW，并设置循环训练epochs次，在每个epoch开始前创建GradScaler（用于混合精度训练），循环遍历训练数据集，将模型输入数据移动到设备上。如果使用混合精度训练，使用autocast进行前向传播计算损失；否则，直接进行前向传播计算损失。如果使用混合精度训练，缩放损失，然后进行反向传播和梯度更新；否则，直接进行反向传播和梯度更新。

在TensorBoard中记录训练损失，在每个epoch后评估模型并保存。



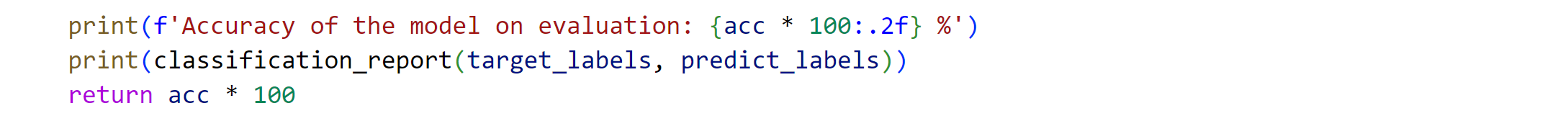




d.定义模型评估函数

设置模型为评估模式，循环遍历测试数据集，将模型输入数据移动到设备上，使用Seqeval库进行前向传播计算输出，解码模型输出，获取预测标签和目标标签，将预测标签和目标标签记录下来，在TensorBoard中记录评估的PR曲线，计算评估准确度并打印分类报告，返回准确度。

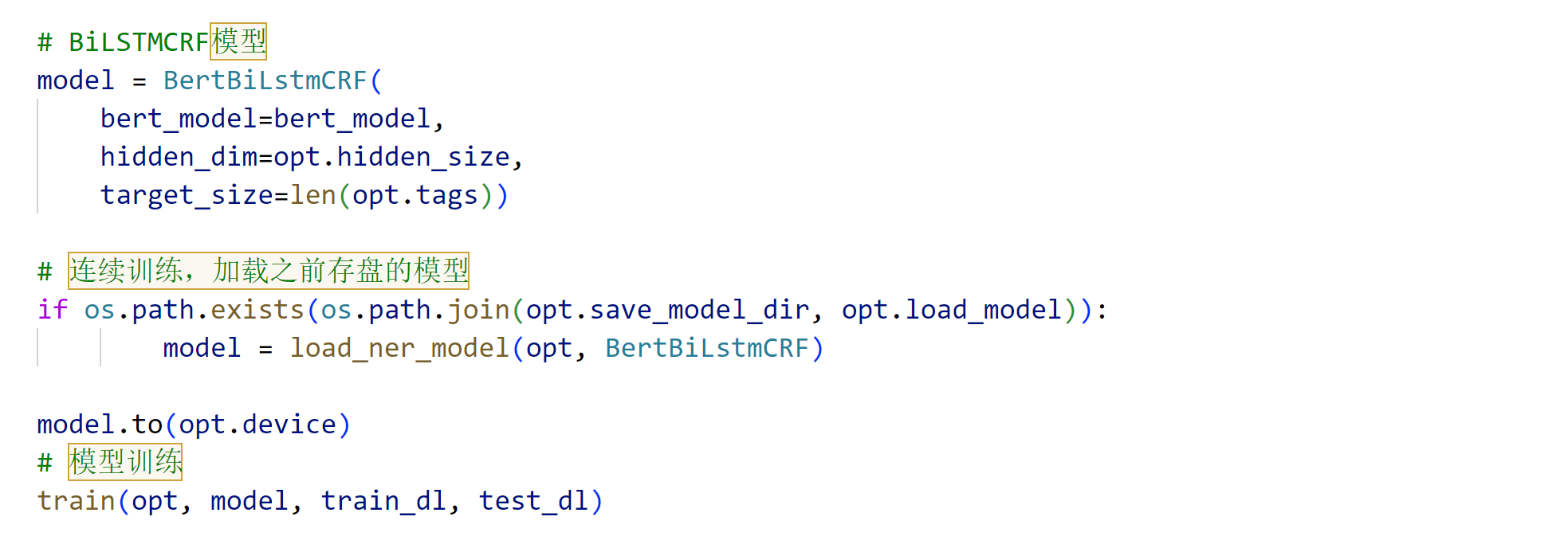




e.主程序入口

初始化损失计数器和PR评估计数器，获取模型参数，加载本地BERT模型和分词器，获取模型训练语料的文件路径，获取训练和测试数据加载器（dataloader），创建BiLSTMCRF模型。如果存在预训练模型，加载之前保存的模型，将模型移动到设备上，调用模型训练函数。





（3）实体识别流程

在实体识别过程中，需要加载本地缓存的BERT模型和Tokenizer，将输入文本转换为模型的输入数据，使用预训练的BERT模型进行实体识别，解码模型输出，提取实体的起始和结束位置，最后将识别到的实体信息以特定格式保存在结果中。

a.导入依赖项

导入必要的Python库和自定义模块，包括处理BERT模型的模块和实用工具函数。



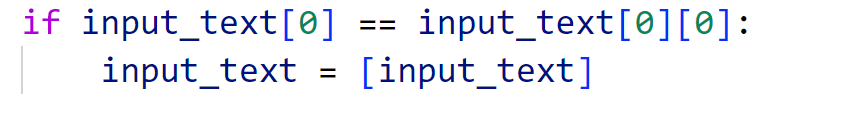
b.定义predict函数

定义了一个名为predict的函数，用于对给定的文本进行实体识别。该函数首先处理输入文本，加载本地缓存的BERTTokenizer和模型，然后执行模型推理并返回实体匹配结果。



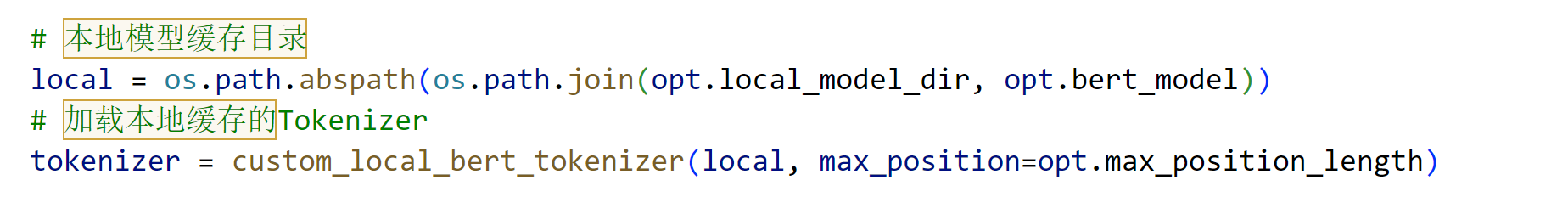
c.处理输入文本

检查输入文本是否为列表，如果不是，则将其转换为单元素的列表。



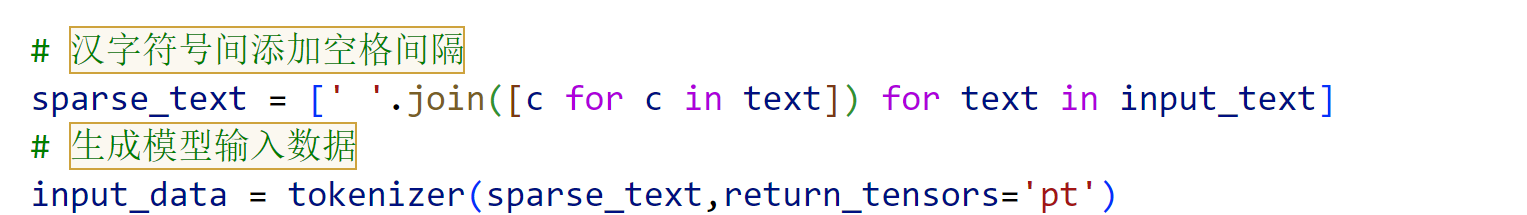
d.加载本地缓存的Tokenizer

构建本地缓存目录路径，并使用该路径加载自定义的BERTTokenizer。



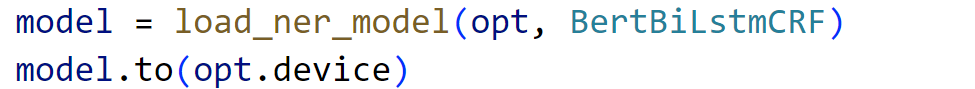
e.文本预处理

将每个汉字之间添加空格，然后使用Tokenizer将处理后的文本转换为模型的输入数据格式。



f.加载模型

使用自定义的函数load\_ner\_model加载BERT+BiLSTM+CRF模型，并将其移动到指定的计算设备（如GPU）上。



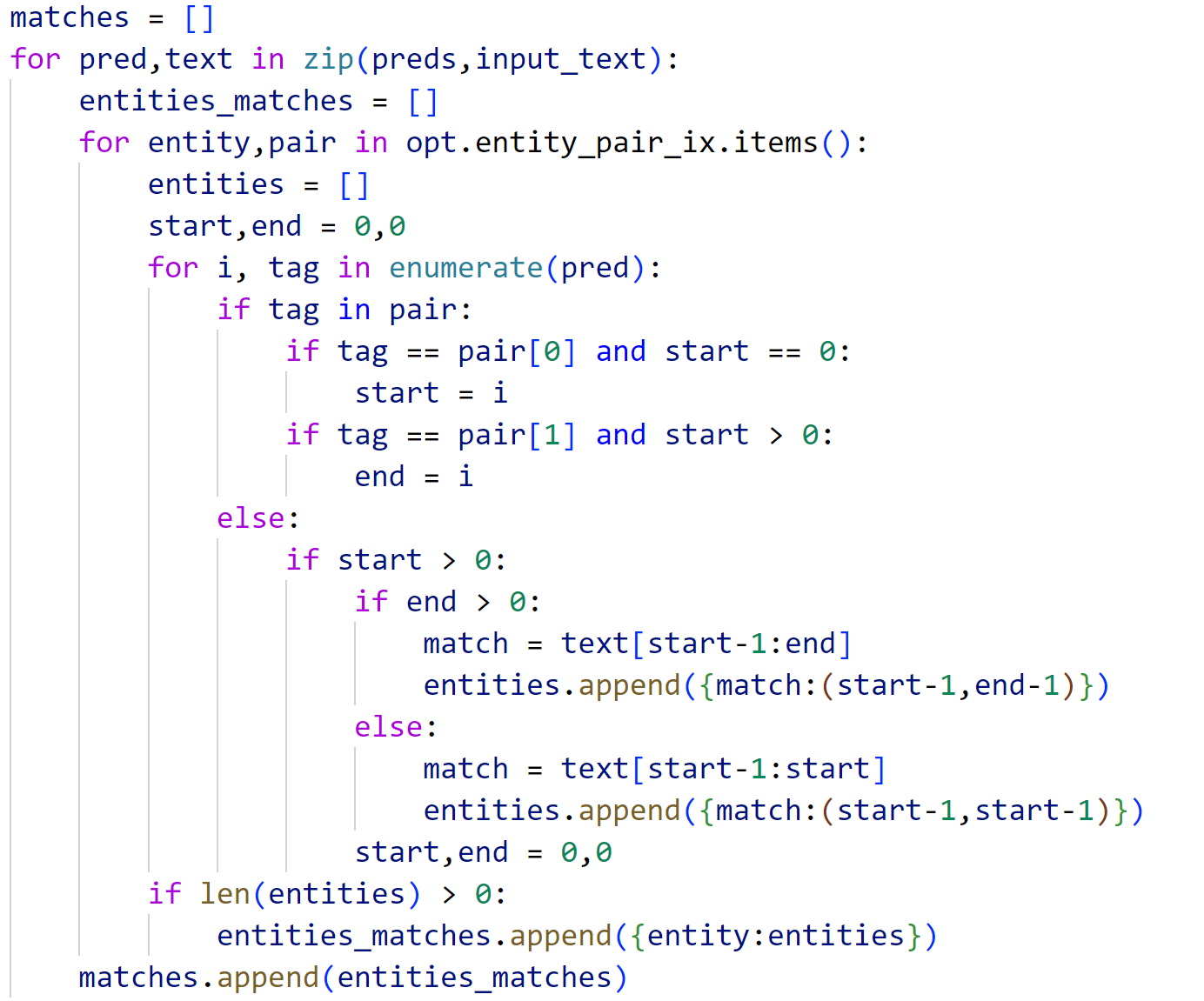
g.模型推理

调用model\_predict函数对输入数据进行模型推理，得到实体标签的预测结果。



h.匹配实体

遍历每个文本的预测结果，并通过匹配实体对的索引，提取匹配的实体文本及其在原文中的位置。



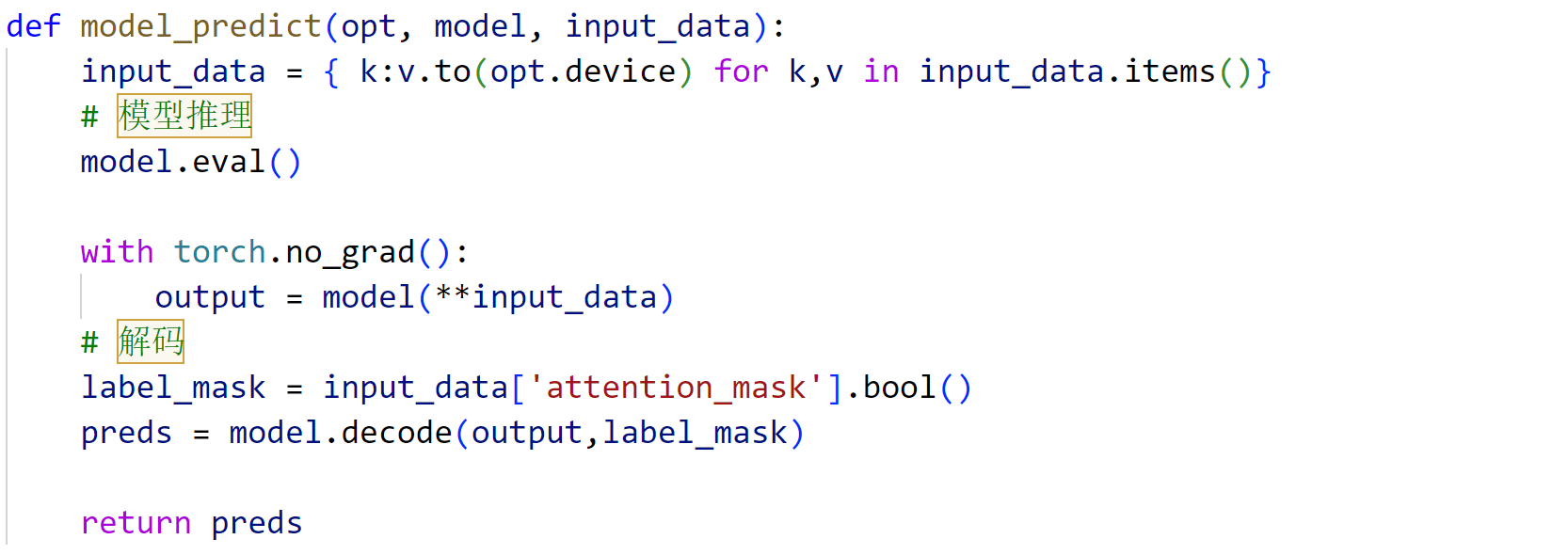
i.返回结果

将匹配的实体结果以特定的格式返回。



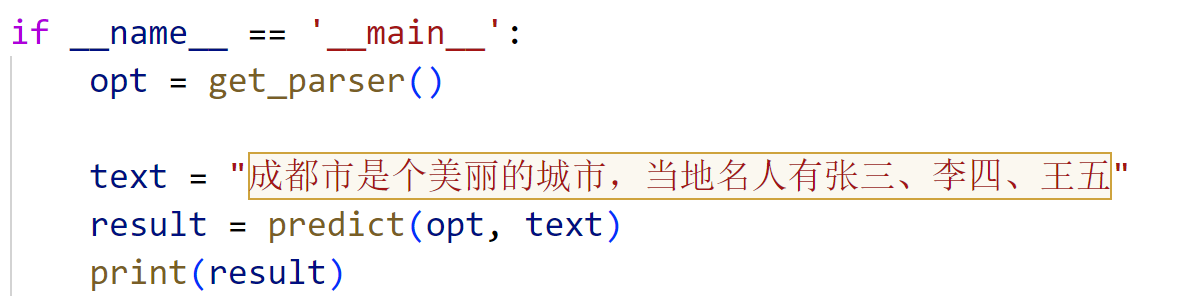
j.定义model\_predict函数

定义了一个函数model\_predict，用于执行模型的推理。该函数首先将输入数据移动到指定的计算设备，然后调用模型进行推理，并解码得到最终的预测结果。



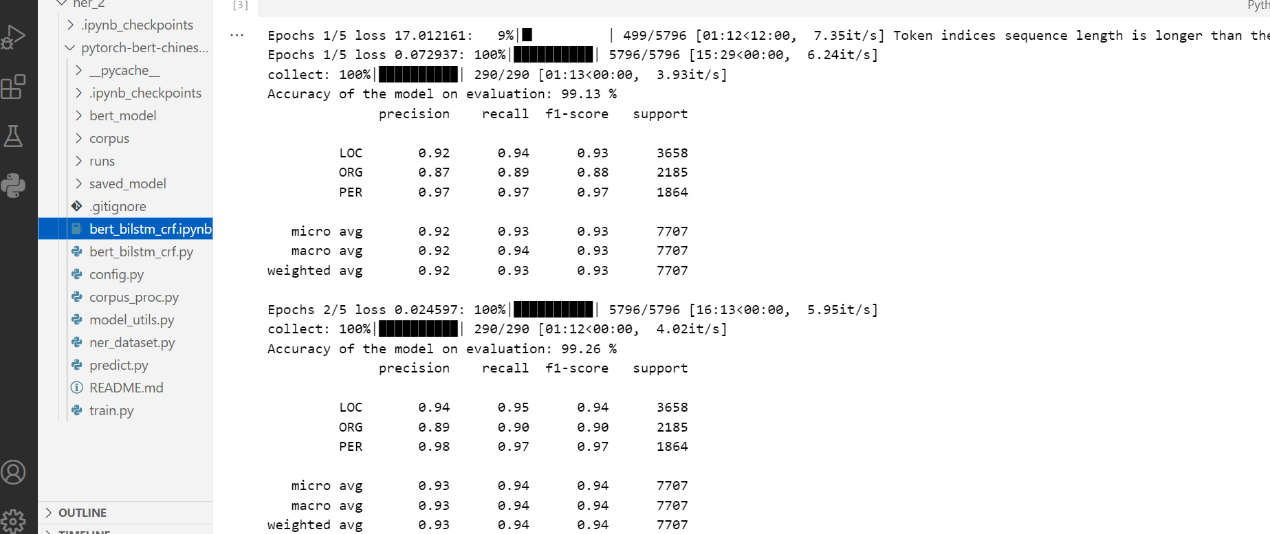
k.执行主程序

在主程序中，首先解析命令行参数，然后给定一个中文文本，调用predict函数进行实体识别，最后打印输出结果。



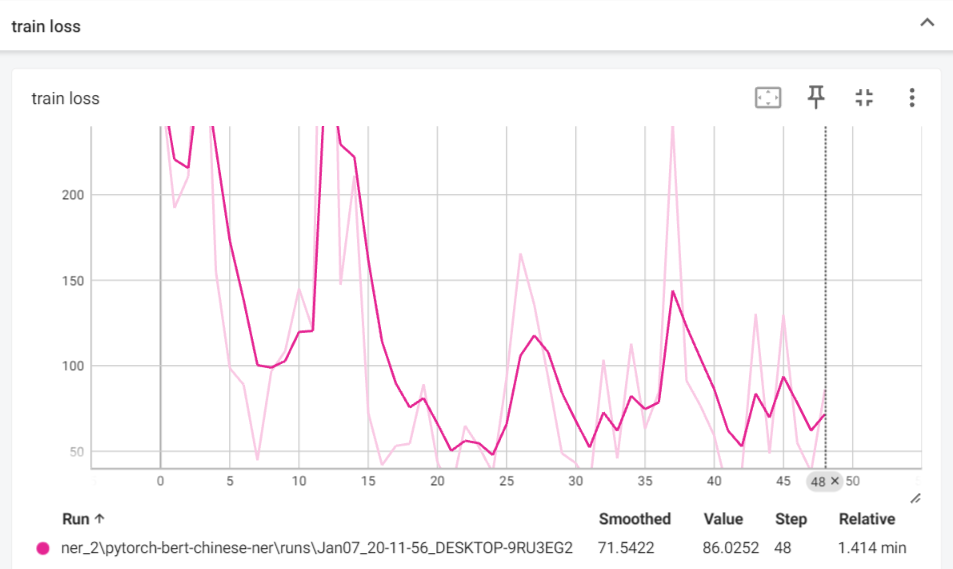
# **结果评测/展示**

4.1模型测试准确率如图：



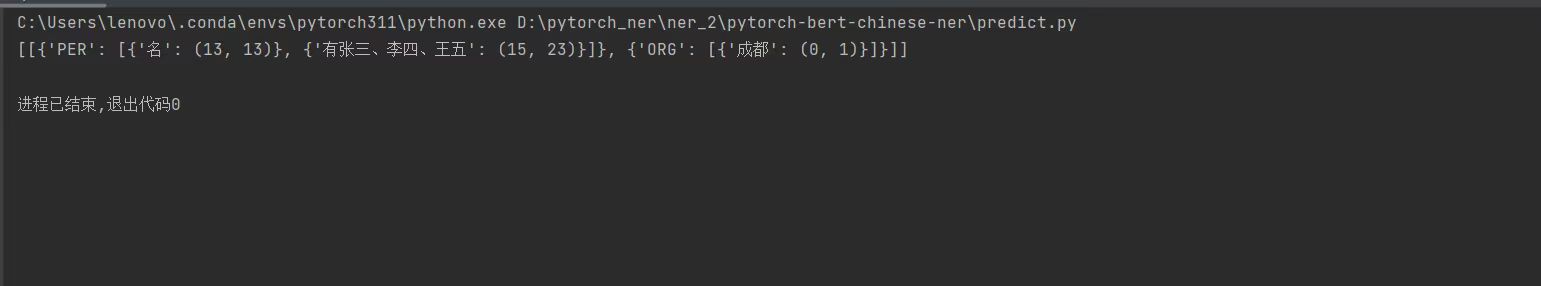
准确率均在99%以上，准确度较高。

4.2TensorBoard观察模型的训练损失



由图可以看出曲线虽然仍有波动，但波动开始趋于平缓，表明模型学习在趋于稳定，且损失曲线正在下降表明模型正在学习并逐渐改善其性能。

4.3命名实体识别(NER)结果展示如下图所示：



结合BERT、双向LSTM和CRF层用于命名实体识别（NER）任务的模型是一种强大的混合方法，每个组件都为任务的成功贡献了重要的特性。同时，使用TorchMetrics库进行性能指标的计算和使用TensorBoard进行训练过程的可视化，可以有效地监控和优化模型性能。

a.BERT层：BERT作为模型的基础，提供了强大的上下文相关的词嵌入。它能够捕捉到词汇在不同上下文中的细微差别，这对于NER任务至关重要。并且BERT的双向特性使得模型能够理解前后文中的复杂语境，从而提高实体识别的准确性。

b.双向LSTM层：双向LSTM能够捕捉文本中的长期依赖关系。它从两个方向处理文本，能够更全面地理解序列中的信息。与BERT结合时，双向LSTM可以进一步提炼和加强BERT提供的上下文信息，使模型对序列数据的处理更加精细。

c.CRF层：CRF层用于序列标注的最后阶段，能够考虑实体标签之间的依赖关系。这对于保持标签序列的一致性和准确性非常重要。通过考虑相邻标签的约束，CRF能够减少标注错误，如将一个实体的一部分标记为另一种实体。

d.TorchMetrics和TensorBoard：使用TorchMetrics库可以准确地计算如准确率、召回率和F1分数等关键性能指标。这些指标对于评估和比较NER模型的性能至关重要。TensorBoard的使用可以有效地可视化训练过程，包括损失函数的变化、性能指标的进展等。这有助于及时发现问题、调整模型参数和改进模型性能。

然而，BERT模型通常需要大量的计算资源进行预训练和fine-tuning。由于其复杂性，BERT在推理时可能比较慢，这在需要快速响应的应用中可能是一个问题。此外，LSTM在处理大规模数据集时的训练速度可能较慢。所以整个模型可能需要较大的计算资源，且模型的复杂性可能导致训练和推理时间较长。

# **总结**

基于深度学习BERT的命名实体识别是一种利用BERT模型进行实体识别的方法。BERT是一种预训练的语言模型，通过在大规模文本数据上进行训练，可以学习到丰富的语义表示。在命名实体识别任务中，BERT可以通过对输入文本进行编码，将每个词语映射为一个向量表示，然后通过分类器来判断每个词语是否属于某个实体类别。

工作总结:

a.数据收集和标注：通过网络爬虫和人工标注等方式，收集并标注了大量的命名实体数据，包括公司名、人名、地名等。

b.数据预处理：对文本数据进行预处理，包括分词、词性标注、字符编码等操作，以便输入到模型中。

c.BERT模型的调用：使用预训练版本的BERT模型，加载权重并将其作为输入编码器，将文本序列转换为上下文感知的词向量表示，可以有效地捕捉上下文信息。

d.BiLSTM层的应用：在BERT的输出上，添加一个BiLSTM层，用于捕捉序列中的上下文信息，并生成更丰富的特征表示。

e.CRF层的引入：在BiLSTM层之后添加一个条件随机场（CRF）层，用于对命名实体进行标签预测和序列标注。CRF层考虑了上下文之间的依赖关系，可以生成最优的标签序列。

f.损失函数和优化算法：使用交叉熵损失函数作为模型的目标函数，并采用梯度下降算法进行模型参数的优化。通过训练模型，可以实现对命名实体的准确识别。

g.模型调整和优化：通过计算模型在验证集和测试集上的准确率、召回率、F1值等指标，对模型的性能进行评估、调整和优化。

局限性分析:

a.训练和推理速度较慢：BERT模型由于参数量庞大，需要大量的计算资源来进行训练和推理。这使得在资源受限的情况下，使用BERT模型可能会面临运行时间较长的问题，对于小规模的数据集可能不太适用。

b.长文本处理限制：BERT模型对输入序列长度有一定的限制。在BERT的原始版本中，输入序列长度不能超过512个标记。当处理较长的文本时，需要进行截断或切分等操作，这可能导致部分信息的丢失。

c.标注数据不足：BERT模型在命名实体识别等任务中通常需要大量的标注数据来进行训练。如果标注数据不足，模型的性能可能会受到限制，可能无法很好地适应特定领域的命名实体识别任务。针对这种情况，可以考虑使用迁移学习或半监督学习等方法来解决数据不足的问题。

d.微调需求：预训练的BERT模型通常是在大规模通用语料上进行的，对于一些特殊领域或特定任务，可能需要进一步进行微调。微调过程需要一定的标注数据，并且可能需要调整超参数以适应特定任务的需求。