Easynlp & Easytext 类间关系分析

一、模块分析

Easynlp:（以官方给出的训练 BERT 模型为例）

ClassficationDataset 数据加载与预处理模块：用于对数据集进行处理，其属性包含了数据集的信息，方法用于处理和转换文本数据。

Application 训练框架模块：这是训练框架的基类模块。

Optimizer 优化器模块：用于更新和管理模型参数。

Trainer 训练控制模块：这个类用于进行训练，其属性包含了训练模型时用到的参数，方法包括设置模型和优化器、从检查点恢复模型训练、设置训练数据加载器等模型训练常用操作。

Evaluator 性能评估模块：用于评估模型的性能。

Exporter 参数转换模块：用于在不同的深度学习框架之间迁移模型。

Losses 损失函数模块：用于定义不同的损失函数。

Easytext:

ConfigFactory 是一个工厂类，用于创建和管理模型的配置。

MetriccTracker 模块主要用于跟踪每一个 epoch 的 metric 记录，保存 best 等指标。

Model 是一个包含预定义结构和参数的实体，用于从输入的数据中进行学习和预测。

Loss 用于评估预测结果和真是值之间的差异。

ModelMetricAdapter 适配器，使 model 和 metric 能一起工作。

Optimizerfactory 是一个工厂类，用于创建和配置优化器。

LRSchedulerfactory 是一个工厂类，用于创建和配置学习率调度器。

Trainer 用于执行训练，评估等任务。

二、类间关系分析

Easynlp 工作流程：（以训练 BERT 模型为例）

初始化 Easynlp 库——解析用户定义的参数——获取预训练模型的路径——创建训练和验证数据集——获取应用模型——创建训练器实例（包含了模型、训练集以及用户定义的参数和评估器）——调用 train 方法进行训练。

```
initialize_easynlp()
args = get_args()
user_defined_parameters = parse_user_defined_parameters(args.user_defined_parameters)
pretrained_model_name_or_path = get_pretrain_model_path(user_defined_parameters.get('pretrain_model_name_or_path', None))

train_dataset = ClassificationDataset(
    pretrained_model_name_or_path=pretrained_model_name_or_path,
    data_file=args.tables.split(",")[0],
    max_seq_length=args.sequence_length,
    input_schema=args.input_schema,
    first_sequence=args.first_sequence,
    second_sequence=args.second_sequence,
    label_name=args.label_name,
    label_enumerate_values=args.label_enumerate_values,
    user_defined_parameters=user_defined_parameters,
    is_training=True)

valid_dataset = ClassificationDataset(
    pretrained_model_name_or_path=pretrained_model_name_or_path,
    data_file=args.tables.split(",")[-1],
    max_seq_length=args.sequence_length,
    input_schema=args.input_schema,
    first_sequence=args.first_sequence,
    second_sequence=args.second_sequence,
    label_name=args.label_name,
    label_enumerate_values=args.label_enumerate_values,
    user_defined_parameters=user_defined_parameters,
    is_training=False)

model = get_application_model(app_name=args.app_name,
    pretrained_model_name_or_path=pretrained_model_name_or_path,
    num_labels=len(valid_dataset.label_enumerate_values),
    user_defined_parameters=user_defined_parameters)

trainer = Trainer(model=model, train_dataset=train_dataset,user_defined_parameters=user_defined_parameters,
    evaluator=get_application_evaluator(app_name=args.app_name, valid_dataset=valid_dataset,user_defined_parameters=user_define
    eval_batch_size=args.micro_batch_size))

trainer.train()
```

图 1 官方给出训练 BERT 模型代码

Easynlp 类间关系：

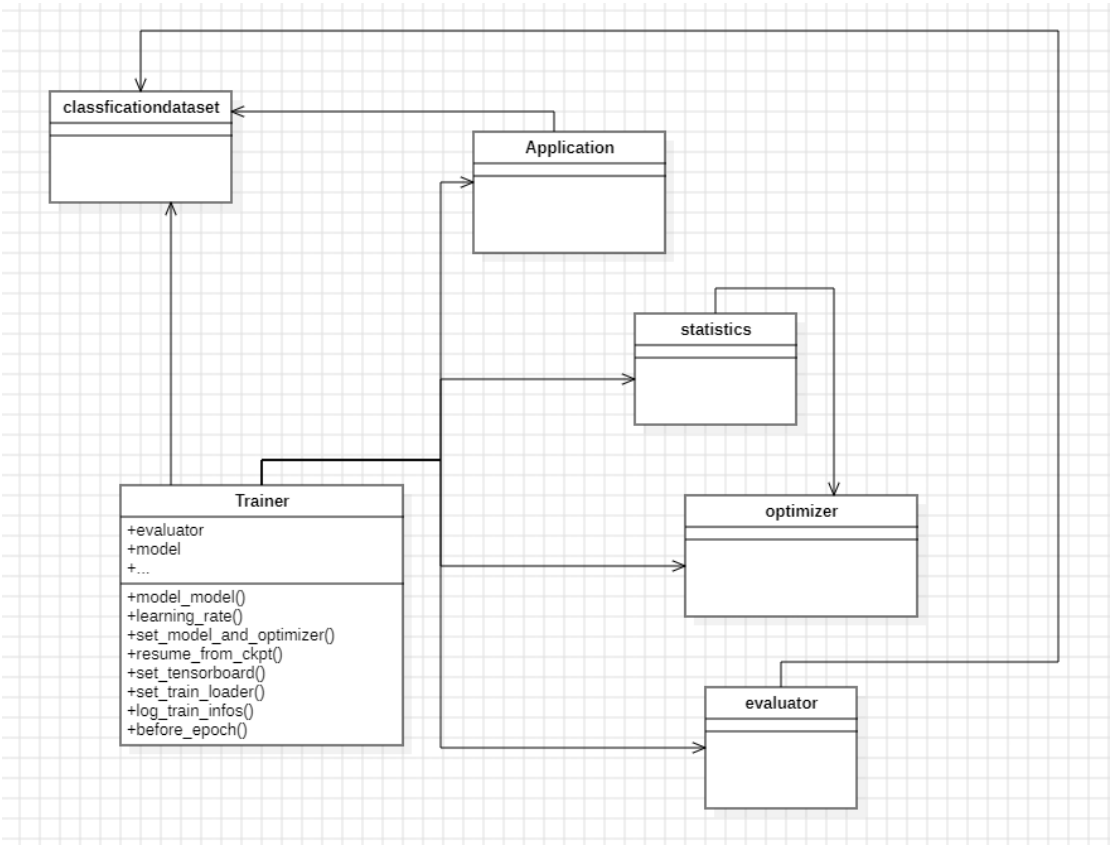

图 2 easynlpuml 图

模型的建立需要验证集中的数据；评估器也需要验证集中的数据进行评估；模型的训练需要训练集的数据。因此这三类都对数据处理类有依赖关系。优化器需要使用到训练过程中 statistics 获取到的数据，因此对其有依赖关系。训练类在训练过程中需要通过调用各种模块完成训练过程，因此它对其它的模块都有依赖关系其中的 Application，classificationdataset 和 evaluator 类都是来源于不同模型的类对基类的继承，这样就实现了在不同的模型上进行训练。

Easytext 工作流程：

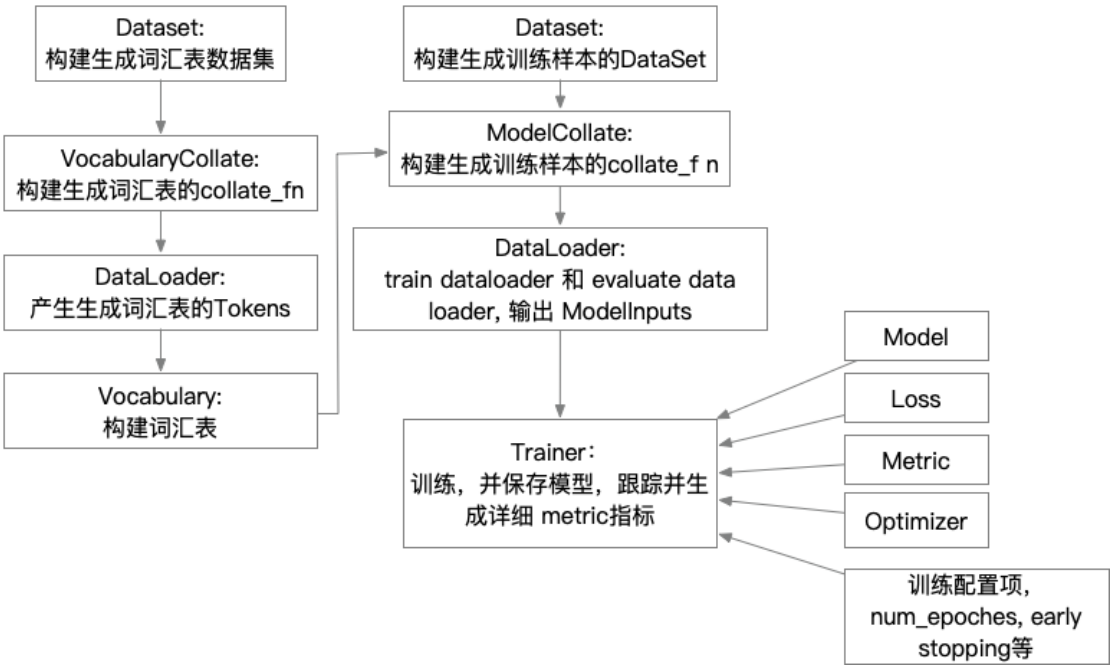

图 3 easytext 工作流程

与 easynlp 类似，也需要先解析用户自定义参数，构建数据集，然后实例化模型和训练器，再进行训练。

Easytext 类间关系：（以训练 ACSA 模型为例）

在 easytext 里，每个模型有一个单独的训练模块，即 Train 类，它接受其它模块处理生成的字典、数据集以及各种参数，并实例化 Loss、metric 等模块最后调用共用的训练器对这些数据进行训练。各个模型也有其对应的数据集与字典等数据，这些数据分别被字典生成模块和数据集生成模块处理后发送给 dataloader，在 train 中进行加载。其它的功能模块如 metric，optimizer 会在 train 类中获取数据及信息并被实例化。

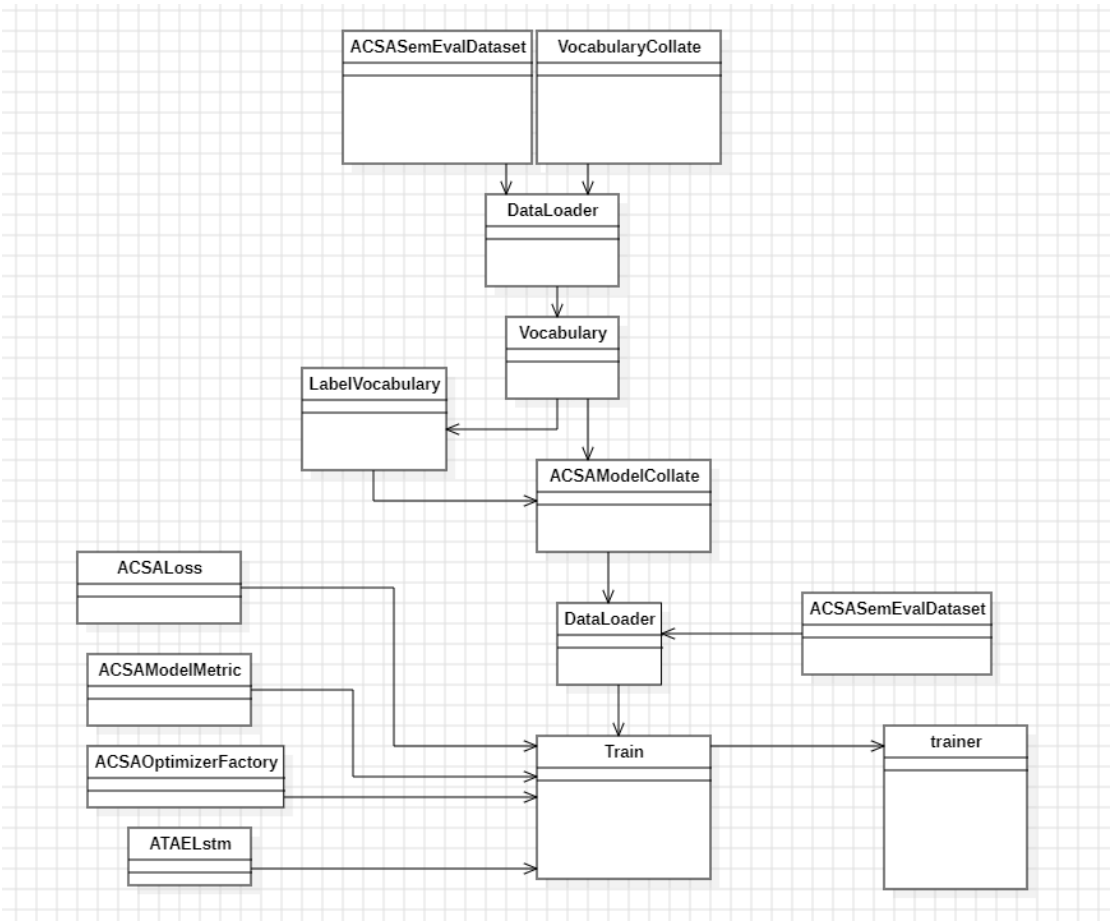

图 4 easytext 类间关系