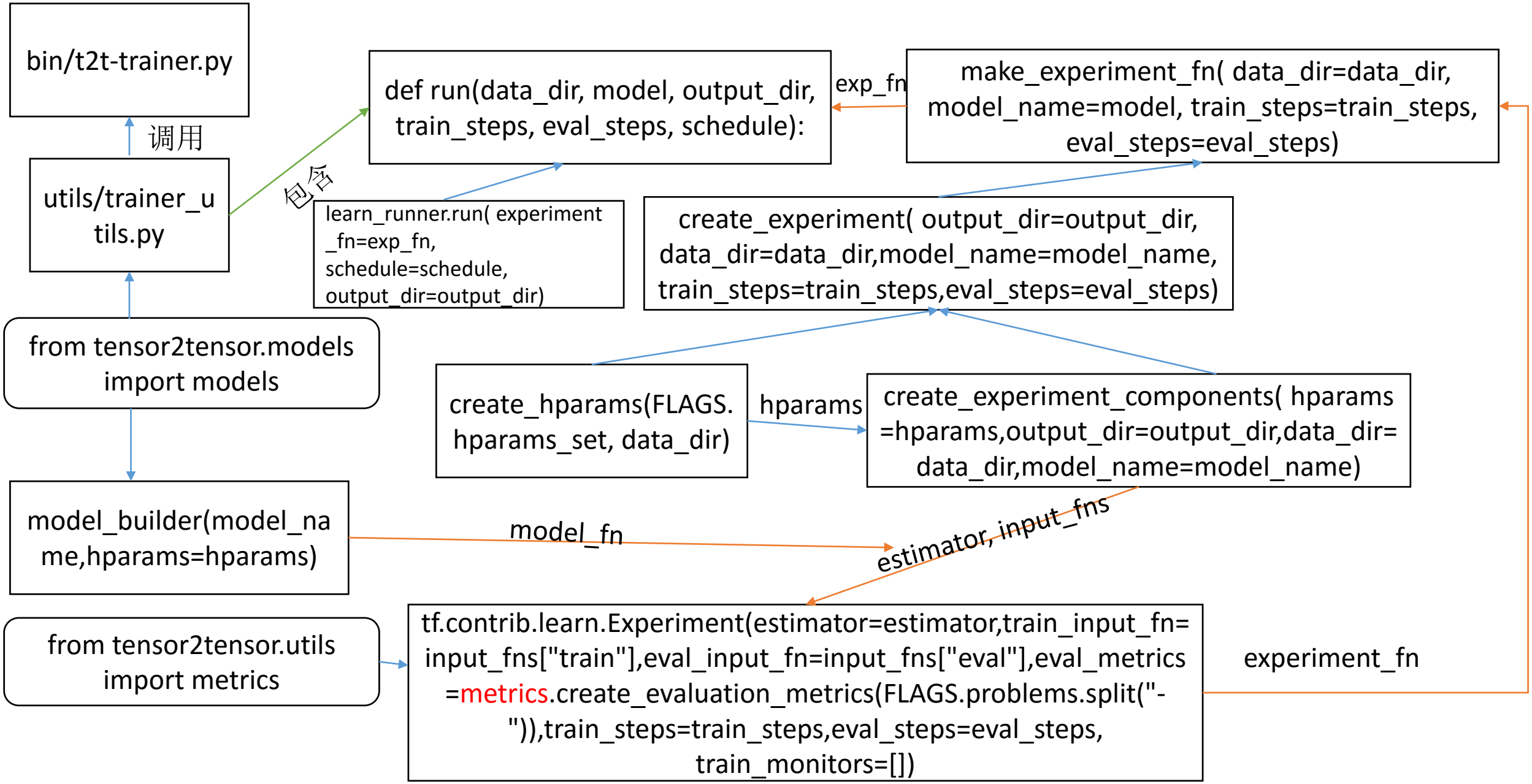
bin/t2t-trainer.py
调用
utils/trainer_utils.py
包含
def run(data_dir, model, output_dir, train_steps, eval_steps, schedule):
exp_fn
make_experiment_fn(data_dir=data_dir, model_name=model, train_steps=train_steps, eval_steps=eval_steps)
learn_runner.run(experiment_fn=exp_fn, schedule=schedule, output_dir=output_dir)
create_experiment(output_dir=output_dir, data_dir=data_dir,model_name=model_name, train_steps=train_steps,eval_steps=eval_steps)
from tensor2tensor.models import models
create_hparams(FLAGS.hparams_set, data_dir)
hparams
create_experiment_components(hparams=hparams,output_dir=output_dir,data_dir=data_dir,model_name=model_name)
model_builder(model_name,hparams=hparams)
model_fn
estimator, input_fns
from tensor2tensor.utils import metrics
tf.contrib.learn.Experiment(estimator=estimator,train_input_fn=input_fns["train"],eval_input_fn=input_fns["eval"],eval_metrics=metrics.create_evaluation_metrics(FLAGS.problems.split("-")),train_steps=train_steps,eval_steps=eval_steps, train_monitors=[])
experiment_fn

```
Estimator=tf.contrib.learn.Estimator(
    model_fn=model_builder(),
    params=model_params,
    config=run_config   )
```

•model_fn：一个功能对象，其中包含所有上述支持训练，评估和预测的逻辑。您负责实施该功能。
•params：将被传递到的超参数（例如，**learning rate**（学习率），**dropout**（退出））的可选指令 **model_fn** 。
•该配置指定如何运行训练和评估，以及如何存出结果。这些配置通过 RunConfig 对象表示，该对象传达 Estimator 需要了解的关于运行模型的环境的所有内容。

```
def model_builder(features, targets, mode, params):
  # Logic to do the following:
  # 1. Configure the model via TensorFlow operations
  # 2. Define the loss function for training/evaluation
  # 3. Define the training operation/optimizer
  # 4. Generate predictions
  # 5. Return predictions/loss/train_op/eval_metric_ops in ModelFnOps object
  return ModelFnOps(mode, predictions, loss, train_op, eval_metric_ops)
```

```
experiment = tf.contrib.learn.Experiment(
    estimator=estimator,  # Estimator
    train_input_fn=train_input_fn,  # First-class function
    eval_input_fn=eval_input_fn,  # First-class function
    train_steps=params.train_steps,  # Minibatch steps
    min_eval_frequency=params.min_eval_frequency,  # Eval frequency
    train_monitors=[train_input_hook],  # Hooks for training
    eval_hooks=[eval_input_hook],  # Hooks for evaluation
    eval_steps=None  # Use evaluation feeder until its empty
)
```

模型函数将输入特征作为参数，相应标签作为张量。它还有一种模式来标记模型是否正在训练、评估或执行推理。模型函数的最后一个参数是超参数的集合，它们与传递给 Estimator 的内容相同。模型函数需要返回一个 EstimatorSpec 对象——它会定义完整的模型。
EstimatorSpec 接受预测，损失，训练和评估几种操作，因此它定义了用于训练，评估和推理的完整模型图。由于 EstimatorSpec 采用常规 TensorFlow Operations，因此我们可以使用像 TF-Slim 这样的框架来定义自己的模型。

Experiment 作为输入：
•一个 Estimator（例如上面定义的那个）。
•训练和评估数据作为第一级函数。这里用到了和前述模型函数相同的概念，通过传递函数而非操作，如有需要，输入图可以被重建。我们会在后面继续讨论这个概念。
•训练和评估钩子（hooks）。这些钩子可以用于监视或保存特定内容，或在图形和会话中进行一些操作。例如，我们将通过操作来帮助初始化数据加载器。
•不同参数解释了训练时间和评估时间。