在线e2e模型训练源代码概要设计说明

# 概述

## 1.1 简介

训练引擎为通用的语音识别任务提供了目前性能最佳的模型训练环境，可以满足项目中大多数任务的需求。针对这些任务的训练，开发者只需要准备好相应的训练数据，便可实现对任务模型的训练。

针对离线语音转录任务，平台提供了基于注意力机制神经网络Transformer的混合CTC/Attention端到端语音识别模型。CTC/Attention模型采用的是编码器-解码器的序列到序列模型，基本思想就是利用两个Transformer网络分别作为编码器和解码器。编码器负责将输入语音特征压缩成指定长度的向量，这个向量就可以看成是这个序列的语义。解码器则负责根据语义向量以及注意力机制的得分生成指定的序列，这个过程也称为解码。此外，混合CTC/Attention模型还在编码器端增加了CTC分支，一方面可以利用CTC损失辅助Attention模型的训练，加速模型的收敛；另一方面，可以通过计算CTC前缀得分，提升模型的识别性能。

针对在线语音转录任务，平台提供了在线版本的Transformer CTC/Attention模型。相比离线模式的Transformer CTC/Attention模型，在线模型采用单调注意力计算的方式以及分段编码的方式，解决了Attention模型对全局信息依赖的问题，极大的压缩了模型的时延且对识别性能的损失极小。与其他在线模型相比，在线Transformer CTC/Attention模型的识别正确率更高，训练速度与解码更快。

## 1.2 读者对象

本文档是针对具有一定python开发的算法人员，了解深度学习的相关算法。本文档可以帮助读者快速了解语音识别模型训练流程，对语音识别形成初步认识。

## 1.3 术语定义

设正确文本字数为N，识别结果文本字数为M，按照识别结果文本与正确文本根据“最小代价匹配”原则运用动态规划算法，得到正确识别字数Mc、删除错误字数D、插入错误字数I，和替换错误字数S，则有：

N=Mc+S+D

M=Mc+S+I

定义以下性能指标：

字正确率： Mcr =(Mc／N)×100％

替代错误率： Ser =(S／N)×100％

插入错误率： Ier =(I／N)×100％

删除错误率： Der =(D／N)×100％

字错误率：WER = ((S+I+D)／N)×100％

## 1.4 参考资料

Joint CTC-Attention based End-to-End Speech Recognition using Multi-task Learning，arXiv:1609.06773

Transformer-Based Online CTC/Attention End-To-End Speech Recognition Architecture，DOI: 10.1109/ICASSP40776.2020.9053165

Joint CTC/attention decoding for end-to-end speech recognition，DOI: 10.18653/v1/P17-1048

https://github.com/espnet/espnet

# 2. 总体设计

训练引擎包括语音识别模型、训练任务构建等模块，其基本训练流程如下图所示：



图2.1 模型训练流程图

## 2.1 语音识别模型

引擎采用的模型为Seq2Seq架构的端到端语音识别模型，包含编码器Encoder和解码器Decoder两部分。主要由卷积神经网络和注意力机制神经网络Transformer两种神经网络结构组成。

卷积神经网络主要由卷积操作和池化操作两部分组成，卷积操作通常是使用一组数量较小的参数权值卷积核，其与输入空间局部相连，并通过滑动对整个输入空间进行分析。由于卷积操作的局部链接的特性，适合于抽取更为精细的特征，可以对原始数据进行分析。池化操作等效于某种形式的下采样过程，以最大池化最为常见。它是将输入的数据划分为若干个矩形区域，对每个子区域输出最大值。该过程会不断地减小数据的空间大小，因此参数的数量和计算量也会下降，这在一定程度上也控制了过拟合。

Transformer是谷歌公司提出的解决序列转换问题的模型，使用全注意力机制的结构代替了长短时记忆神经网络中的递归结构，并在机器翻译任务中取得了巨大的进展。目前，Transformer模型在语音识别领域也有了广泛的应用。首先，使用注意力机制对信号的时序关系进行建模，避免了循环神经网络中存在的遗忘现象。其次，Transformer使用的是非递归结构，可以在GPU上实现高速的并行计算。最后，Transformer使用多个注意力头，使得Transformer可以关注到语音信号中不同维度的信息。

## 2.2 训练任务的构建

训练引擎将训练过程中每一项需求抽象为一个任务（Task）。对于每一个Task，包含数据中心（DataCenter），训练器（Trainer），监控器（Observer）以及验证器（Valider）四部分，对于一个训练任务，各部分的功能与关系如下图所示。

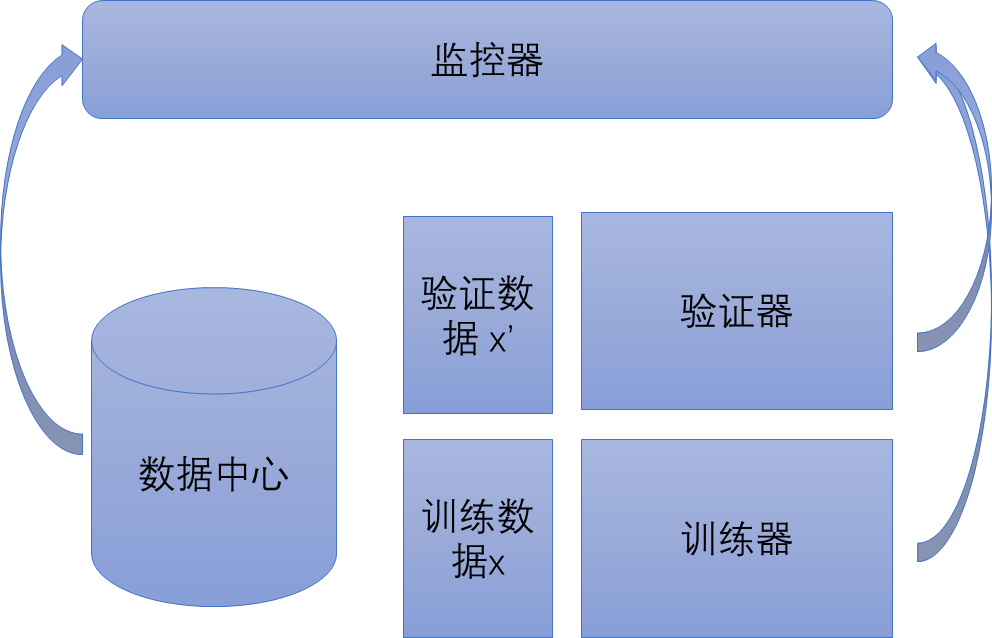


图2.2 训练任务的构建

通过对任务类的管理，实现训练数据的读取、训练、验证等功能。

# 3. 可靠性设计

模型训练平台考虑了输入数据的复杂性与多样性，为了确保输入训练数据的有效性，在数据加载部分对数据进行过滤，过滤太短、太长的对模型训练无任何帮助的数据，同时能够确保模型训练正常运行，防止显存占用过高导致崩溃问题出现。

模型训练平台增加了异常判断功能及相关的日志输出，可以迅速定位问题，并对模型训练进程和性能进行评估，在多个任务上进行了测试验证，保证模型训练的稳定性和可靠性。

# 4. 扩展性设计

模型训练平台提供了标准的语音识别模板任务，同时对底层接口和数据类型进行封装，提高平台的可扩展性，方便开发者使用。

模型训练平台采用模块化设计，方便用户对模型结构、损失函数等进行扩展。

模型训练平台支持多卡进行训练，充分利用GPU资源，提高模型训练速度。

# 5. 安全性设计

模型训练平台采用自研平台，在运行平台上实现自主可控。

对模型训练框架进行了整体设计，并搭建多元化底层框架，可以实现大规模语音识别模型的训练，并优化训练策略和训练方法，以实现安全可控的智能声学信息处理平台。

# 6. 易用性设计

模型训练平台采用简单易用的接口设计，方便开发人员使用。模型训练提供了训练、评估、测试的脚本，用户可以直接调用上述三个脚本对模型进行训练、评估和测试，方便用户评估训练模型性能。

模型训练平台支持tensorboard日志输出功能，通过可视化的方式，对模型训练过程中的loss、acc进行监督，及时调整模型训练策略。

模型训练平台通过外部配置文件控制模型训练参数，并提供详细的配置参数说明，方便用户修改模型训练参数。