哈爾濱工業大學

实验报告

实 验（二）

题　　目　连续语音识别

专　　业　人工智能

学　　号　2021112845

班　　级　2103601

学　　生　张智雄

指 导 教 师　郑铁然

实 验 地 点　格物楼 213

实 验 日 期　2024.4.14

计算机科学与技术学院

实验二:连续语音识别

1、 实验内容或者文献情况介绍

1.1 自动语音识别的背景

自动语音识别技术(Automatic Speech Recognition, ASR)是一种人机交互技术，旨在将人类语音输入转换为文本或命令的过程。该技术利用计算机算法和模型，对语音信号进行分析和处理，以识别和理解说话者所表达的内容，并将其转换为可识别和可操作的文本形式。

语音识别技术在诸多领域具有广泛应用，包括但不限于语音助手、语音搜索、自动转写以及语音命令控制等。

语音识别系统一般有**模型训练、语音识别**两个过程，包含了特征提取、声学模型、发音词典、语言模型、语音解码五个部分。

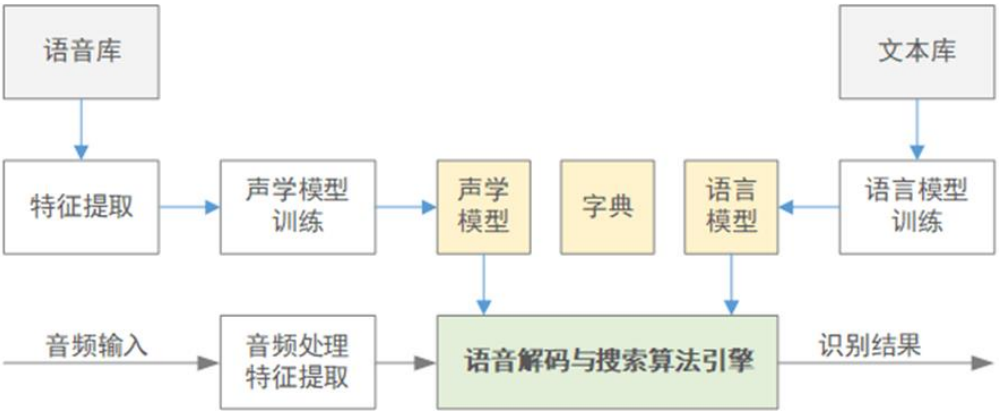


图 1 语音识别系统

1.1.1 特征提取

在语音识别过程中，声学信号（即音频）被转换成数字化数据后，其波形在时间上呈现出复杂的震动变化。由于每个音频文件的波形都各不相同，因此要使计算机能够理解和区分不同音频的内容，需要从声学参数中提取出具有区分性的特征。

常用的声学特征提取方法包括时域到频域的转换、频谱分析等。在语音识别系统中，常用的声学特征包括梅尔频率倒谱系数（MFCC）、感知线性预测（PLP）、Fbank（Filter-bank）等。

提取出声学特征后，通常还需要进行 CMVN（Cepstral Mean and Variance Normalization，倒谱均值方差归一化）处理，以确保各个特征参数的形式一致性。

1.1.2 声学模型

声学模型是语音识别系统中用来描述语音特征和识别结果之间关系的数学模

型。它基于声学特征对音频进行建模，使得每个音频可以用一个函数或公式来描述。在声学模型的训练过程中，常使用动态时间规整（DTW）、隐马尔可夫模型（HMM）、高斯混合模型（GMM）、深度神经网络（DNN）等算法。

声学模型训练通常从单音素（mono-phone）的 GMM 训练开始，然后逐步迭代至三音素（Triphone）的 GMM 训练，最终使用三音素 DNN 进行训练。这些模型的迭代训练过程旨在不断提高模型的准确性和性能。

1.1.3 发声词典

发声词典字或词和其发言音素的对照表文件，其功能为将语音信号映射到对应的文本或命令，帮助系统理解用户的语音输入。

1.1.4 语言模型

语言模型是一种统计模型，用于根据语言学规律计算特定字符串（句子）出现的概率。常用的是 N-gram 模型，它通过对大量文本数据进行训练，以了解词语之间的搭配和频率，从而预测输入语音的可能性。在语音识别中，语言模型帮助系统解码出现概率最高的词序组合，以更准确地转录或理解用户的语音输入。

1.1.5 语音解码

语音解码根据输入的声学特征，从声学模型、发音词典模型和语言模型构建的图网络中，找到最优路径，即最可能的词序列，并且估计其相关得分。解码过程中，通常采用 WFST（带权重的有限状态转换器）解码机制，将动态知识源编译成静态网络，以加快解码速度。

因此，语音解码实质上是在图的网络中寻找最优路径的过程，从声学特征到词序列的映射过程。

1.2 评价指标

词错率（Word Error Rate, WER）是一项用于评价 ASR 性能的重要指标，用来评价预测文本与标准文本之间错误率，因此词错率最大的特点是越小越好。

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C}$$

1.3 实验内容

本实验需完成连续语音识别任务，包括以下步骤：

a) 找到一种连续语音识别模型；

b) 进行算法复现；

c) 进行识别测试。

2、 基于 Kaldi2 框架的 TDNN-CTC 模型

2.1 Kaldi2 框架

Kaldi2 是一个基于 PyTorch 的开源语音识别工具库，旨在构建自动语音识别（ASR）系统。它提供了灵活而强大的 API，用于构建和训练大规模的 ASR 模型。Kaldi2 的核心算法基于有限状态自动机（Finite State Automaton），使用了 KWS 搜索网络和转移深度神经网络（TDNN）来实现 ASR 任务。

Kaldi 提供了丰富的功能和模块，包括声学模型训练、声学特征提取、解码器、语言模型集成等。它支持多种声学模型，包括基于高斯混合模型（GMM）和深度神经网络（DNN）的模型。Kaldi 还支持多种解码器，如基于 HMM（Hidden Markov Model）和 WFST（Weighted Finite State Transducer）的解码器，以及支持多种语言模型的集成。该工具库具有以下主要特点和实现细节：

a) **强调通用算法和 recipe。**通用算法指的是像线性变换这样的算法，而不是那种只能针对某种特定语音的算法。

b) **尽量避免把简单问题复杂化。**当构建一个大型的语音识别系统的时候很容易出现很少使用的代码。为了避免这点，每一个命令行工具都只针对有限的情况。

c) **Kaldi 是容易理解的。**虽然 Kaldi 是一个庞大的系统。但是对于每一个工具，可读性都较强。

d) **Kaldi 是容易复用和重构的。**各个模块尽量松耦合。这就意味着一个头文件需要 include 的头文件尽可能少。

2.2 TDNN-CTC 模型

TDNN-CTC （Time Delay Neural Network - Connectionist Temporal Classification）是一种用于语音识别的声学模型架构。它通过结合时间延迟神经网络（TDNN）和连接主义时间分类（CTC）来实现端到端的语音识别。

TDNN（Time-Delay Neural Network，时延神经网络）是一种用于处理时序数据的神经网络结构，其核心思想是利用滑动窗口来捕获输入数据中的时间相关性。具体来说，为了实现时移不变性，将一组延迟添加到输入（音频文件，图像等），使得数据在不同的时间点被表示，随后在每个时间窗口内应用神经网络层进行特征提取和处理。这些不同时间窗口的输出结果被连接在一起，形成了最终的输出。

TDNN 的关键之处在于其卷积操作的时间维度。传统的卷积神经网络（CNN）通常只在空间维度上进行卷积操作，而 TDNN 在时间维度上也进行了卷积，因此能够捕捉到输入数据中的时间依赖性和序列信息。

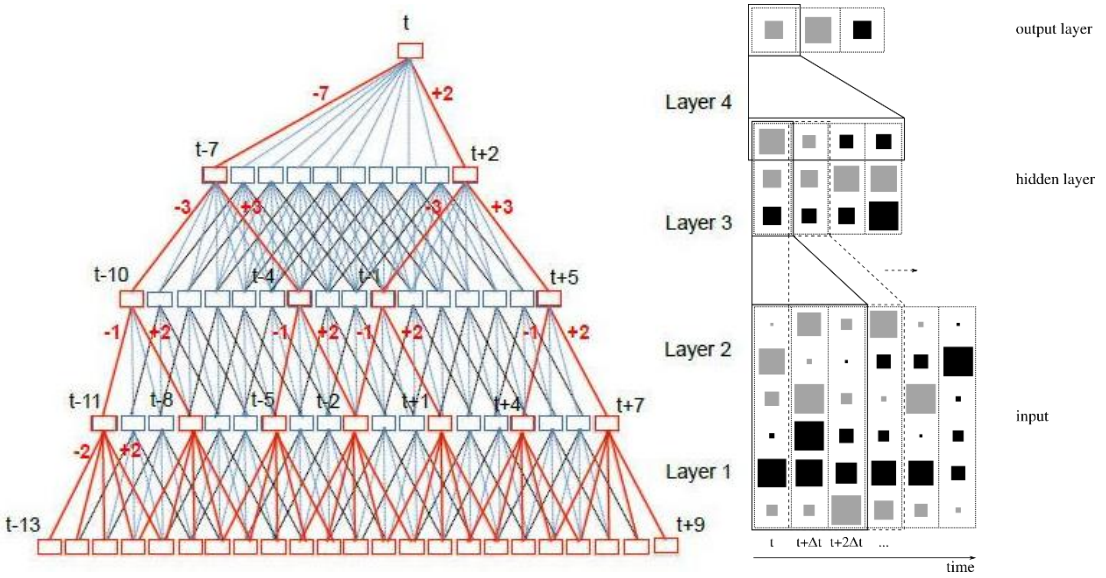


图 2 TCNN结构示意图

而 CTC 是一种端到端的序列学习方法，用于解决序列标注问题，例如语音识别中的音素识别。它考虑了输入序列和输出序列之间的对齐问题，不需要显式地提供对齐信息，而是通过学习对齐关系来进行训练。

CTC 的目标是学习一个映射函数，将输入序列映射到输出序列的概率分布。这个函数通过最大化真实标签序列的概率来进行训练，而忽略了对齐过程。

设输入序列为$\boldsymbol{O} = \{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_T\}$, 对应的标签序列为$\boldsymbol{L} = \{l_1, \ldots, l_M\}$，有$(\boldsymbol{O}, \boldsymbol{L}) \in \boldsymbol{S}$, 则损失函数应为如下交叉熵形式：

$$\mathrm{CTC}(\theta) = \sum_{(\boldsymbol{O},\boldsymbol{L})\in\boldsymbol{S}} -log\ P(\boldsymbol{L} \mid \boldsymbol{O}, \theta)$$

而 TDNN-CTC 的训练过程和解码过程如下：

- 训练过程：在训练阶段，TDNN-CTC 使用带有 CTC 损失函数的反向传播算法来优化网络参数。通过最小化 CTC 损失函数，网络被训练来使得真实标签序列的概率最大化。
- 解码过程：在推理阶段，TDNN-CTC 使用贪婪搜索或束搜索等方法来从输出概率分布中解码出最可能的标签序列。这些标签序列被用作最终的识别结果。

TDNN-CTC 结合了 TDNN 结构的时序特征提取和 CTC 损失函数的端到端学习能力，成功地实现了语音识别任务的端到端处理。通过自动学习时序信息和对齐关系，避免了传统系统中繁琐的特征工程，使得语音识别模型更加简洁高效。

2.3 Yesno 数据集

由于个人笔记本的计算资源有限，本实验采用较为轻量的 YesNo 数据集在 TDNN-CTC 模型上进行训练测试。

YesNo 数据集是一个常用的语音识别数据集，由一系列短语音片段组成，每个片段包含一个人说出“yes”或“no”的录音。该数据集主要用于二分类任务，即识别口语中的是或否（yes 或 no）。

2.4 实验过程及结果

本项目的模型来自于 https://github.com/k2-fsa/icefall 中的 yesno 项目。

配置好环境后，调用 prepare.sh 脚本以准备 yesno 数据集，而后执行./tdnn/train.py 脚本在 TDNN-CTC 模型上进行训练。

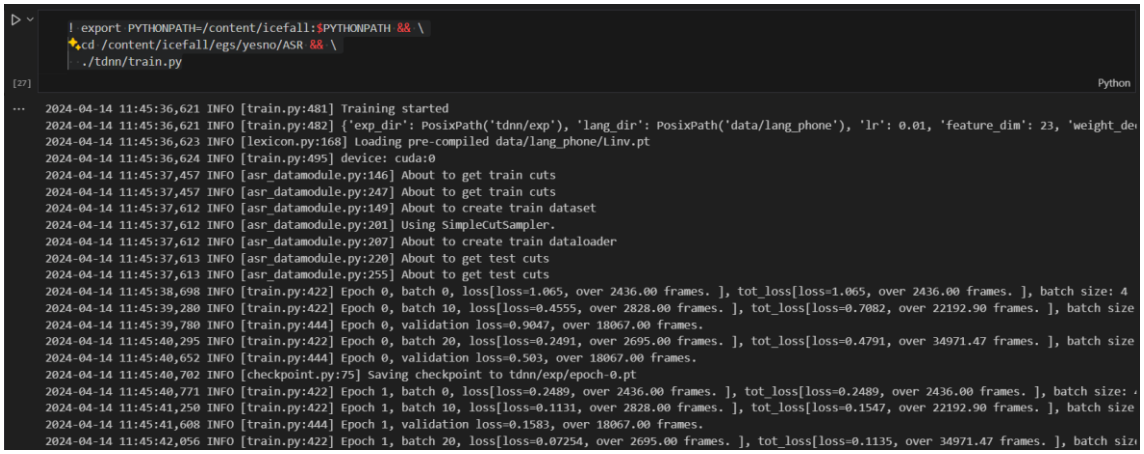


图 3 训练过程

训练完成后调用./tdnn/decode.py 进行解码和结果评估，测得$WER = 0.42\%$，[1 / 240, 0 ins, 1 del, 0 sub]，结果如下：

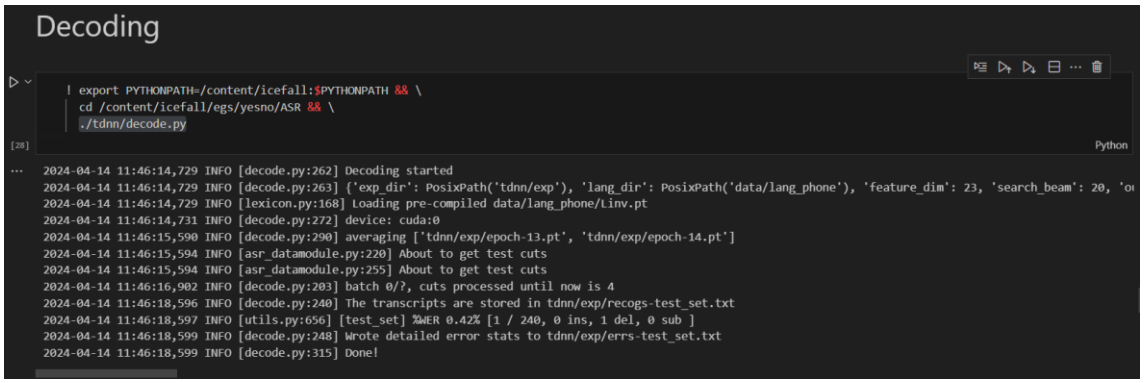
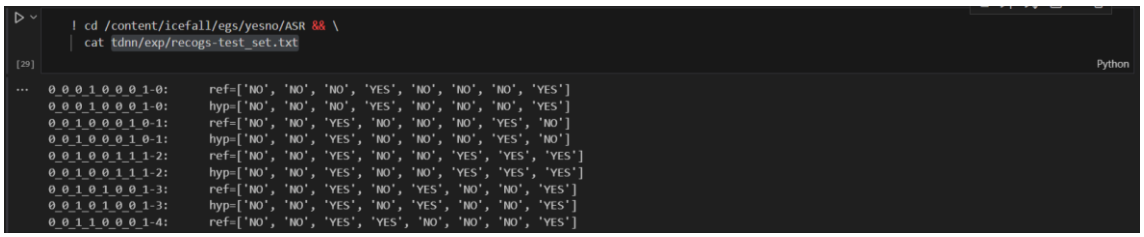


图 4 解码测试结果

具体的结果也保存到 tdnn/exp/recogs-test_set.txt 中可以查看如下：



3、 基于 Transformers 框架的 Wav2Vec 2.0 模型

3.1 Transformers 框架

Transformers 是由 Hugging Face 开发的一个 NLP 包，支持加载目前绝大部分的预训练模型，支持 100 多种语言的文本分类、信息抽取、问答、摘要、翻译、文本生成。

3.2 Wav2Vec 2.0 模型

wav2vec 直接输入音频信号（raw audio），输出音频信号的表征向量（embedding），该表征向量可以作为下游任务的音频特征进行输入。

wav2vec 模型主要分为编码网络(encoder network)、量化部分(quantization)和上下文网络(context network)两个部分。

- encoder 网络($f: \mathcal{X} \rightarrow \mathcal{Z}$)：5 层卷积层，kernel size 分别是 10,8,4,4,4 依次减小，stride 分别是 5,4,2,2,2 依次减小；encoder 网络的主要作用是将原始音频信号转换为高层次的特征表示，起到信息提纯的效果。
- 量化网络($f: \mathcal{Z} \rightarrow \mathcal{Q}$)：使用乘积量化对编码器网络输出的向量$\mathcal{Z}$进行离散化，首先将$\mathcal{Z}$分解为多个子空间，每个码本通过 Gumbel-Softmax 或聚类寻找与输入向量最相似的条目，最后将所有子空间的输出拼接成一个离散化后的向量。
- context 网络($f: \mathcal{Q} \rightarrow \mathcal{C}$)：由 1.0 版本的卷积层改为 transformer 结构；context 网络的主要作用是学习音频特征的时序信息和上下文关系。

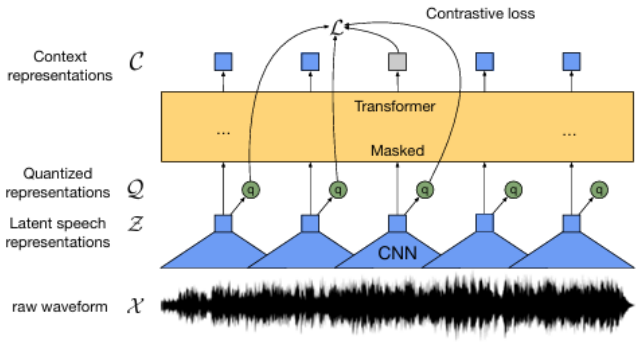


Figure 1: Illustration of our framework which jointly learns contextualized speech representations and an inventory of discretized speech units.

图 5 Wav2Vec2.0模型架构

encoder 网络使用了因果卷积，通过特殊的 padding 方法使卷积输出的每个元素仅依赖于其左侧（过去）和当前的输入元素，而不依赖右侧（未来）的输入元素。

而量化过程的主要作用是起到了对特征向量压缩去冗的效果，同时通过在各子空间内的聚类，使特征的鲁棒性更强，不易受少量扰动的影响。

context 网络把 BERT 取代了 CPC，把 BERT 和编码器、量化器结合在一起，使用 BERT 的 Masked 掩码机制实现 Seq2Seq 的训练。

wav2vec 2.0 的损失函数包括对比损失和多样性损失两个部分，最终的损失值由这两部分损失加权得到。

对比损失： $\mathcal{L}_m = -\log \frac{\exp(sim(\boldsymbol{c}_t, \boldsymbol{q}_t)/\kappa)}{\sum_{\tilde{\boldsymbol{q}} \sim \boldsymbol{Q}_t} \exp(sim(\boldsymbol{c}_t, \tilde{\boldsymbol{q}})/\kappa)}$

多样性损失： $\mathcal{L}_d = \frac{1}{GV}\sum_{g=1}^{G} -H(\bar{p}_g) = \frac{1}{GV}\sum_{g=1}^{G}\sum_{v=1}^{V} \bar{p}_{g,v} \log \bar{p}_{g,v}$，其中 $p_{g,v} = \frac{\exp(l_{g,v}+n_v)/\tau}{\sum_{k=1}^{V} \exp(l_{g,k}+n_k)/\tau}$

其中$\boldsymbol{c}_t$是 context 网络的输出，$\boldsymbol{q}_t$是量化后的特征向量，，其中输入到 context 网络的特征向量z经过随机 mask 处理，而输入到量化模块的向量则不经过 mask 处理；$\tilde{\boldsymbol{q}}$表示的是负样本，κ是温度参数。

$p_{g,v}$表示的是第g个子空间使用第v个聚类中心的概率，多样性损失函数的作用是鼓励模型尽量等可能性的使用每个子空间的每个聚类中心。

3.3 数据集

本实验采用 LibriSpeech，包含文本和语音的有声读物数据集，由 Vassil Panayotov 编写的大约 1000 小时的 16kHz 读取英语演讲的语料库。数据来源于 LibriVox 项目的阅读有声读物，并经过细致的细分和一致。

音频为英语。有两个配置：清晰（clean）和其他（other）。根据在不同数据集上训练的模型的转录的 WER，对语料库中的演讲者进行了排名，并且大致上被分为了两组，“清晰”的演讲者被指定为"clean"，而较高 WER 的演讲者则被指定为"other"。本实验均采用"clean"的子集进行推理和测试。

3.4 实验过程及结果

3.1.1 推理

首先使用 Wav2Vec 2.0 模型进行推理。主要包括以下步骤：

1) 导入必要的库和模块，包括 soundfile、torch、datasets 和 transformers 并加载预训练的 Wav2Vec2 模型和处理器（processor）。

2) 使用 load_dataset 函数加载 ibrispeech_asr_dummy 数据集的验证集（这是 LibriSpeech 数据集的一个截断版本。）。

3) 从数据集中读取音频文件获取音频数据和采样率，并使用处理器（processor）对音频进行预处理，将其转换为模型所需的输入值，并转换为 PyTorch 张量。

4) 使用模型进行推理，得到预测的 logits，并通过取 logits 的 argmax 得到预测的标签序列。使用处理器（processor）对预测的标签序列进行解码，得到最终的预测文本。

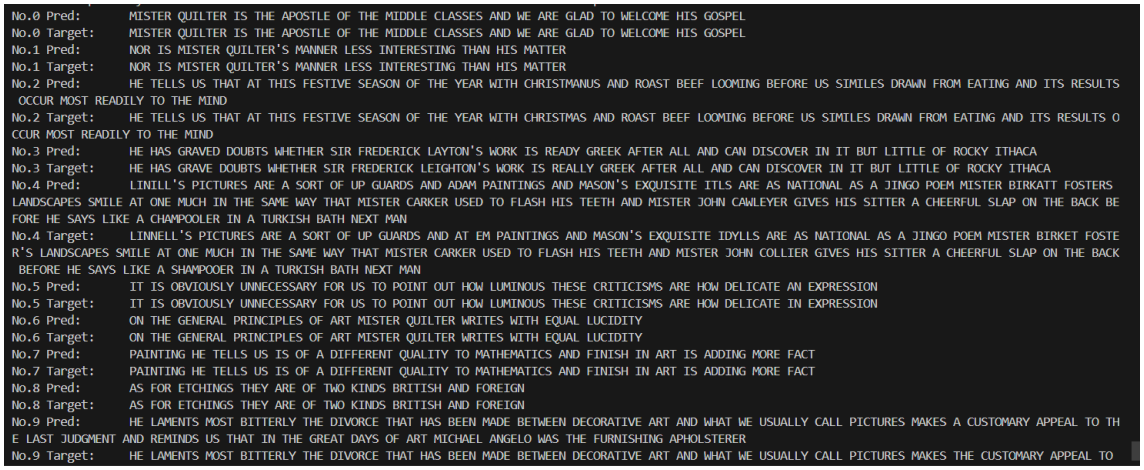


图 6 推理结果

3.1.2 评估

接下来使用 WER 指标评测预训练的 Wav2Vec2 模型，同样包括以下步骤：

1) 导入必要的库和模块，包括 soundfile、torch、datasets 和 transformers 等并加载预训练的 Wav2Vec2 模型和处理器（processor）。

2) 使用 load_dataset 函数加载 LibriSpeech 数据集的测试集。

3) 定义 map_to_pred 函数，先将音频数据预处理为模型所需的输入值，然后通过模型进行推理，得到预测的标签序列，并使用处理器进行解码，得到最终的预测文本。

4) 使用 map 方法对整个数据集应用 map_to_pred 函数，得到预测结果，对每个样本进行预测和计算，得到平均 WER

最终得到的平均 WER 为 0.038250165024441105

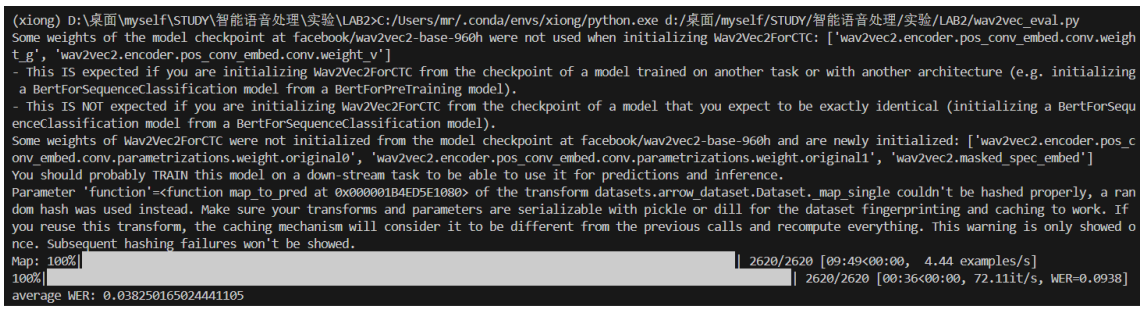


图 7 评估结果

4、 实验心得体会

通过本次实验，我们学习了两种不同的语音识别模型：基于 Kaldi2 框架的 TDNN-CTC 模型和基于 Transformers 框架的 Wav2Vec 2.0 模型。

通过复现和测试这两种模型，掌握了模型的构建和训练过程，加深了理解语音识别技术的理解，对连续语音识别的背景和相关技术有了更深入的了解，为未来在语音处理任务中的应用打下基础。