# 论文/实验记录

## Abstract

因为语音增强后的语音不能再ASR中直接使用，所以我们想到一种方式，再语音增强部分得到一个粗糙的语音，可以不用听懂，但是能直接再ASR模型中使用~~~

## 1、Introduction

## 2、Main net~~~

### 思路一：语音增强这块将语音增强和得到粗糙语音合并

语音增强部分，基于Attention训练一个语音增强模型。网络输入，带噪语音；网络输出，经过attention的变换后的语音。网络部分学习clean的fbank特征，得到的粗糙语音，用带噪语音的fbank去学习纯净语音的fbank特征

ASR模型librispeech数据集的结果

Baseline：用语音增强后直接得到的语音，作为ASR训练好的模型的测试集的输入

SE\_Fbank+ASR：用语音增强（se\_fbank）得到的粗糙语音，作为ASR训练好的测试集的输入

### 思路二：

直接将经过fbank网络调整后的fbank特征保存到mat文件中，作为ASR的测试输入，避免恢复语音出现的问题

### 思路三：

SE\_Fbank+ASR：先得到语音增强后的语音，再经过fbank的Attention网络（纯净语音VS 语音增强后的语音）让mix学习clean的feat,保存到mat文件中。

注意：SE部分训练集验证集用wisj+ TIMITS，py问价为test2

Clean/dev\_dir=/data01/AuFast/Pan\_dataset/Exp3\_SE/T\_dataset

Test\_dir=/data01/AuFast/origin\_dataset/dataset/LibriSpeech/test\_dataset/SE

Noisy\_dir = '/data01/AuFast/Pan\_dataset/Exp3\_SE/data3/Test\_NoiseX-92-16000'

训练集：（14494）：TIMITS（TIMIT-TRAIN）+ wsj0（si\_tr\_s）；信噪比：-5，0

验证集：（1857）：wsj0（si\_et\_05+ si\_dt\_05）；信噪比：-5，0

## 3、Experimental Setup

### 3.1思路一的实验配置

#### 3.1.1 ASR数据集：

训练集、验证集测试集都是librispeech

#### 3.1.2 Baseline:SE+ASR

1. SE

Clean/dev\_dir=/data01/AuFast/Pan\_dataset/Exp3\_SE/T\_dataset

Test\_dir=/data01/AuFast/origin\_dataset/dataset/LibriSpeech/test\_dataset/SE

Noisy\_dir = '/data01/AuFast/Pan\_dataset/Exp3\_SE/data3/Test\_NoiseX-92-16000'

训练集：（14494）：TIMITS（TIMIT-TRAIN）+ wsj0（si\_tr\_s）；信噪比：-5，0

验证集：（1857）：wsj0（si\_et\_05+ si\_dt\_05）；信噪比：-5，0

测试集：（2620）：Librispeech（test-clean）；信噪比：-5，0，5

#### 3.1.3 SE\_Fbank+ASR

训练集测试集和验证集同上

### 3.2 思路二的实验配置

#### 3.2.1 ASR数据集：

训练集、验证集测试集都是librispeech

#### 3.2.2 Baseline:SE+ASR

Librispeech

训练集验证集，信噪比0，-5

测试集，信噪比0，-5，5

数据在：/data01/AuFast/Pan\_dataset/SE\_asr/test1/se\_dataset

生成flac文件，保存在：/data01/AuFast/Pan\_dataset/SE\_asr/test1/se\_asr/gen\_test\_flac

保存为：se\_asr.pt

#### 3.2.3 SE\_Fbank+ASR

训练集验证集测试集同上，生成mat文件

保存在：/data01/AuFast/Pan\_dataset/SE\_asr/test1/se\_fbank\_asr/gen\_test\_mat/

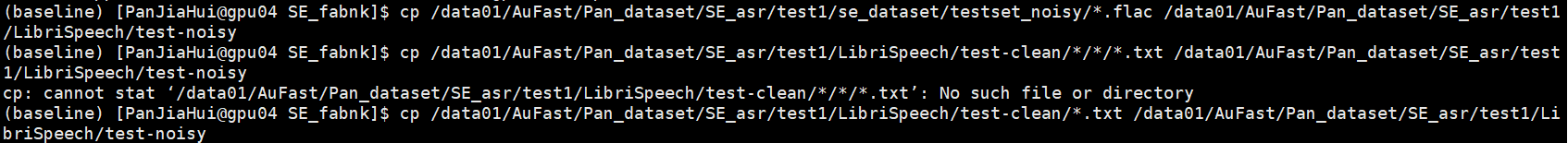
保存为：se\_fbank\_asr\_test2.pt

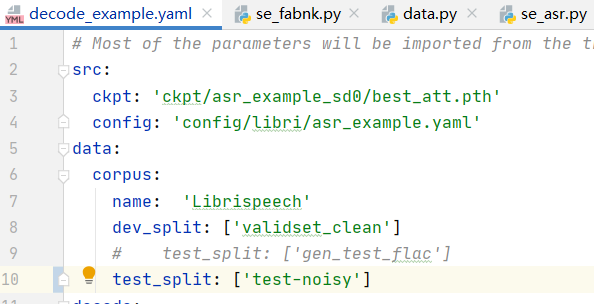
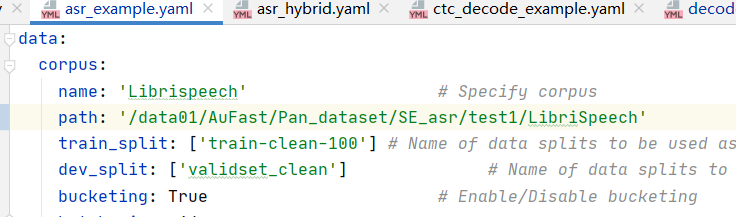
### 3.3 思路三的实验配置

#### 3.3.1 ASR数据集：

Asr1:用libri的测试集的clean进行评估

路径：/data01/AuFast/Pan\_dataset/SE\_asr/test1/LibriSpeech/test-noisy





Result保存在：D:\Study\DeepLearning\Code\dataset\log\asr\test2\_asr\_test\_noisy

Asr2:用libri的测试集的noisy进行评估

路径：/data01/AuFast/Pan\_dataset/SE\_asr/test2/se\_fbank/testset\_clean

Result保存在：D:\Study\DeepLearning\Code\dataset\log\asr\test2\_asr\_test\_clean

#### 3.3.2 Baseline:SE+ASR

训练集，验证集为wsj0+timits：/data01/AuFast/Pan\_dataset/Exp3\_SE/T\_dataset

测试目的是生成flac文件，在SE\_fbank中去使用

保存在：test2.pt

测试集有三部分：分别是librispeech的训练集，测试集和数据集

源数据路径：/data01/AuFast/Pan\_dataset/SE\_asr/test1/se\_dataset

生成路径：/data01/AuFast/Pan\_dataset/SE\_asr/test2/se/gen\_Libri

把生成的数据复制到：/data01/AuFast/Pan\_dataset/SE\_asr/test2/se\_fbank，作为noisy

再次对训练集和验证集进行训练

#### 3.3.3 SE\_Fbank+ASR

对上一步骤的训练集和验证集进行训练，

训练测试路径：/data01/AuFast/Pan\_dataset/SE\_asr/test2/se\_fbank

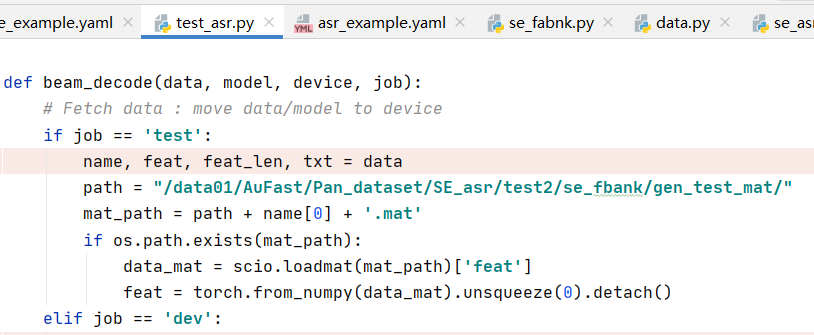
保存在：se\_fbank\_asr\_test2.pt

对测试集进行操作，把测试集生成mat文件，保存在：

/data01/AuFast/Pan\_dataset/SE\_asr/test2/se\_fbank/gen\_test\_mat/

复制到在ASR中使用

路径：/data01/AuFast/Pan\_dataset/SE\_asr/test2/se\_fbank/gen\_test\_mat/



## 4、Results

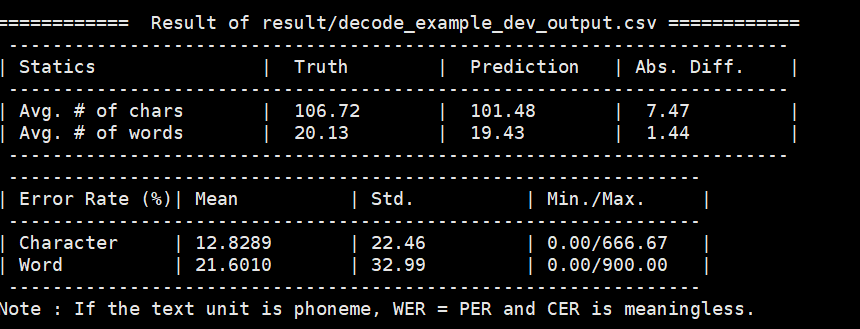
### 4.1思路一的实验结果

#### 4.1.1 ASR

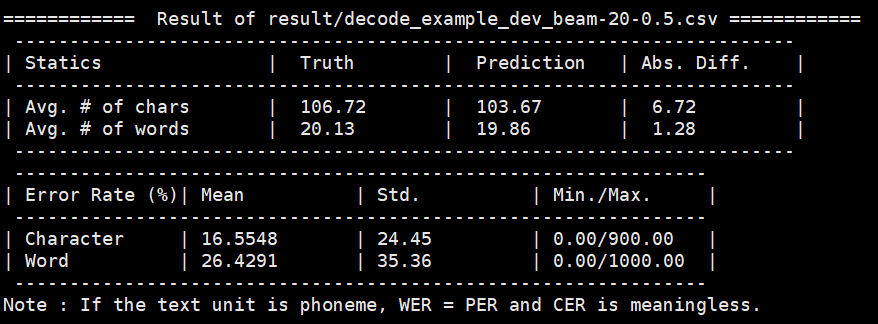
用Librispeech测试集（test-clean）的结果

1. 验证集：

decode\_example\_dev\_output.csv：

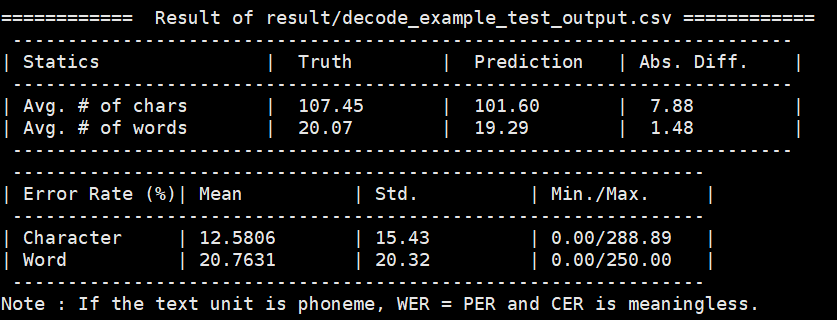


decode\_example\_dev\_beam-20-0.5.csv：

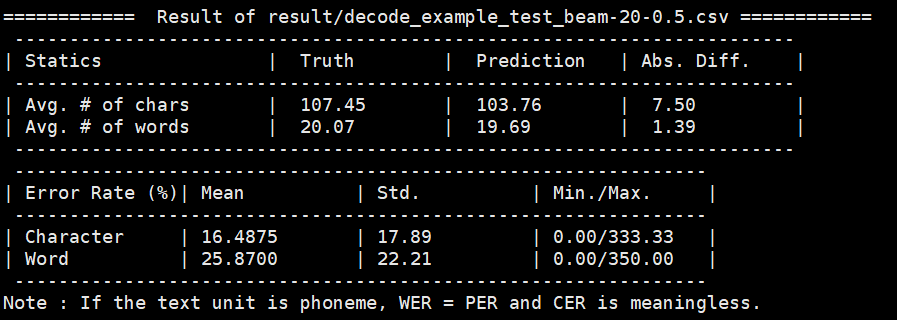


1. 测试集：

decode\_example\_test\_output.csv：



decode\_example\_test\_beam-20-0.5.csv:



#### 4.1.2 Baseline：SE+ASR



1. 生成降噪后的语音：

/data01/AuFast/origin\_dataset/dataset/LibriSpeech/test\_dataset/SE/testset\_clean

1. 测试路径：

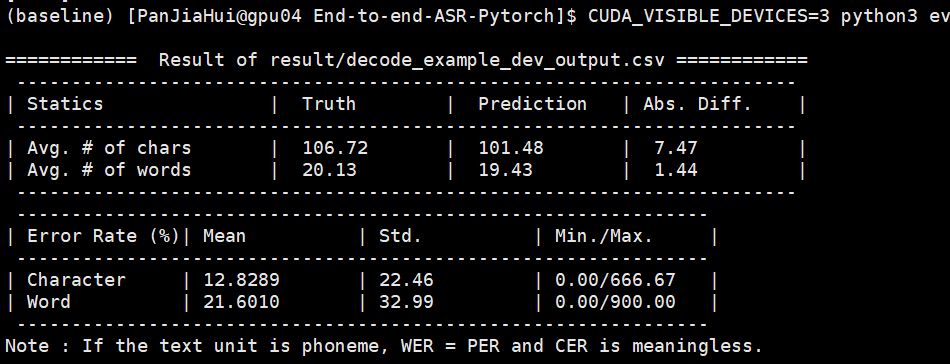
复制到：/data01/AuFast/origin\_dataset/dataset/LibriSpeech/se\_test

1. SE降噪后的语音+ASR结果：

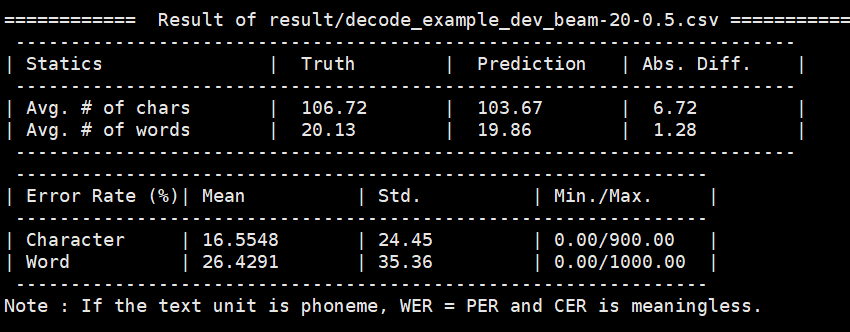
CUDA\_VISIBLE\_DEVICES=2 python3 main.py --config config/libri/decode\_example.yaml --test --njobs 8

1. 验证集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_dev\_output.csv

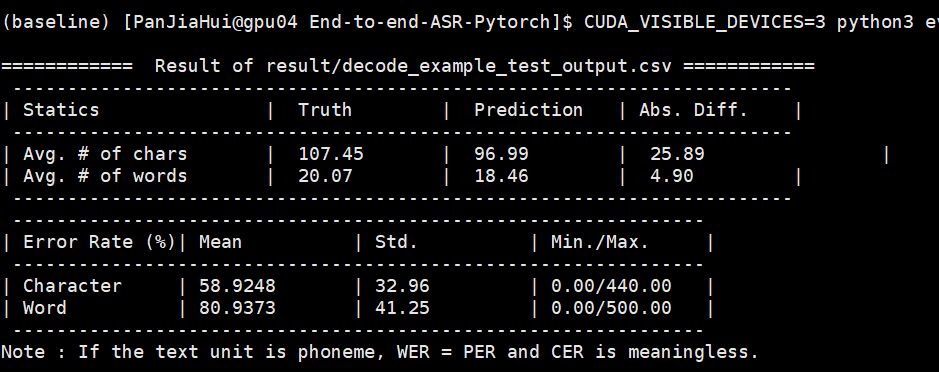


decode\_example\_dev\_beam-20-0.5.csv：

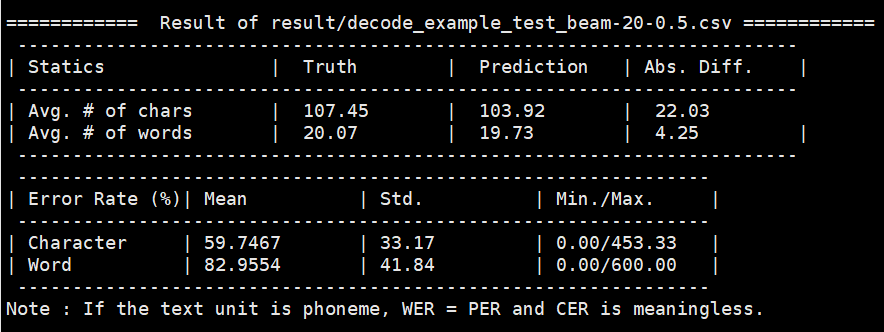


1. 测试集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_test\_output.csv



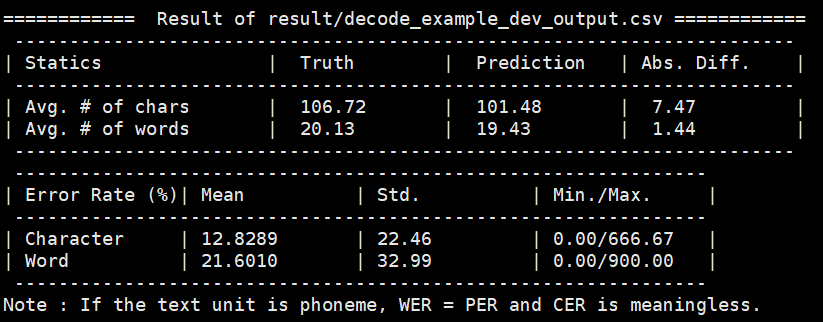
decode\_example\_test\_beam-20-0.5.csv



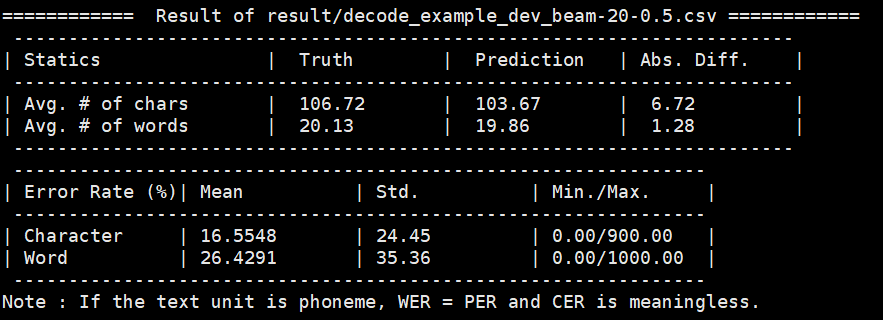
#### 4.1.3 SE\_Fbank+ASR

##### 验证集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_dev\_output.csv

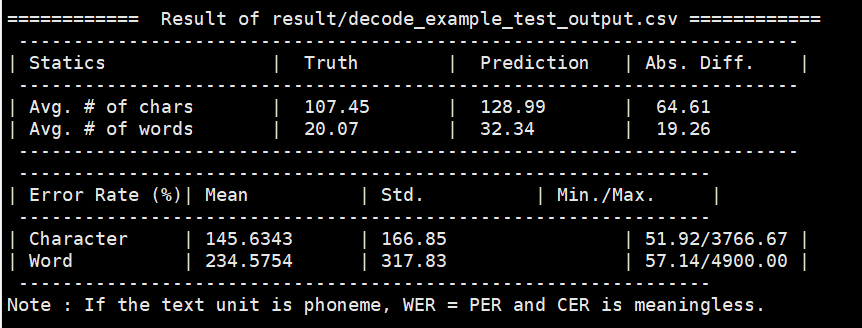


decode\_example\_dev\_beam-20-0.5.csv：

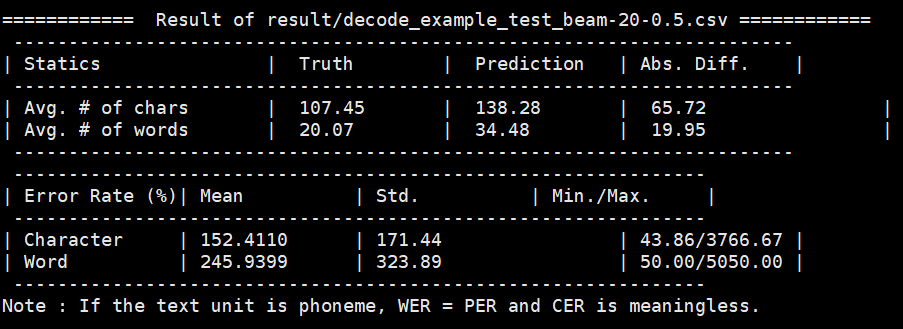


##### 测试集结果：

decode\_example\_test\_output.csv



decode\_example\_test\_beam-20-0.5.csv



#### 4.1.4 分析

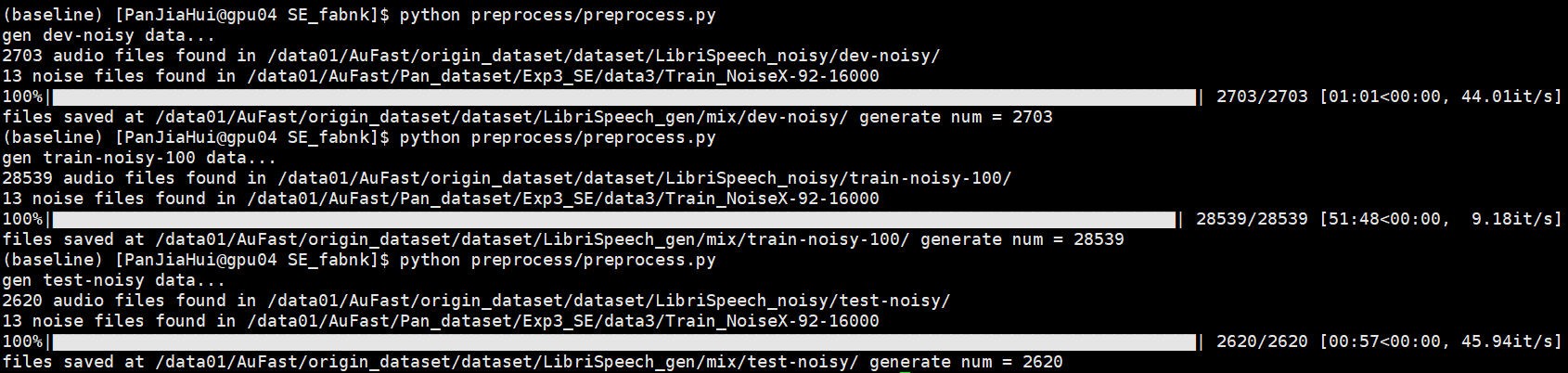
结果特别差，我认为问题所在：

1. Se\_fbank的时候，经过网络的语音恢复有问题
2. 网络有问题

改进：

如果想得到粗糙的语音，就无法避开语音恢复这个步骤，所以还是应该解决这个问题

#### 4.1.5 改进

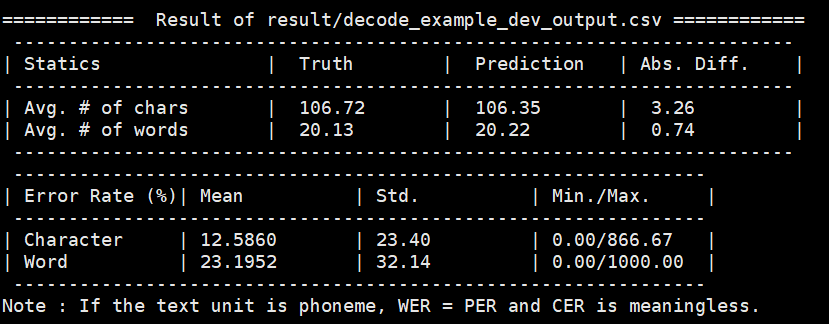


### 4.2思路二的实验结果

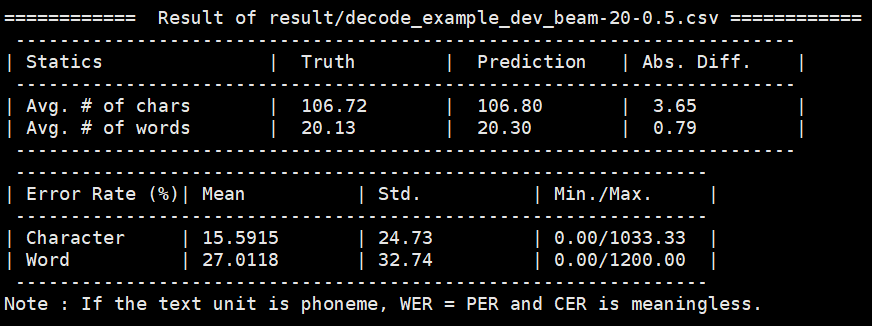
#### 4.2.1 ASR

##### 验证集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_dev\_output.csv

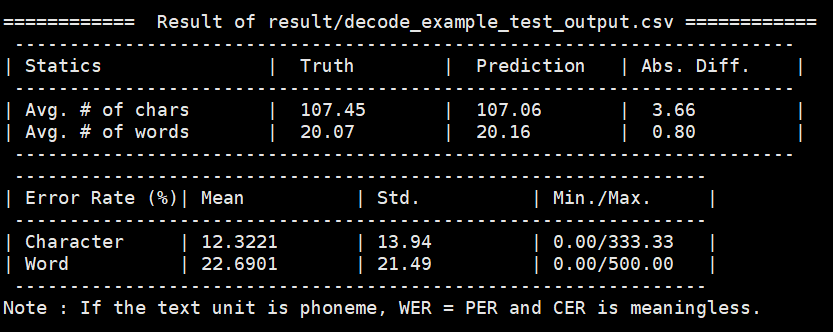


decode\_example\_dev\_beam-20-0.5.csv：

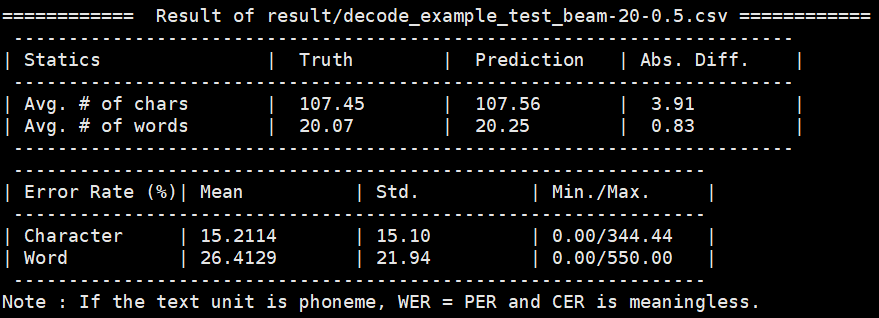


##### 测试集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_test\_output.csv



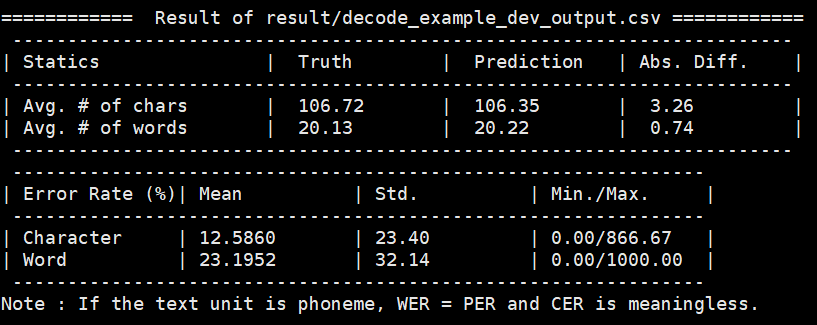
decode\_example\_test\_beam-20-0.5.csv：



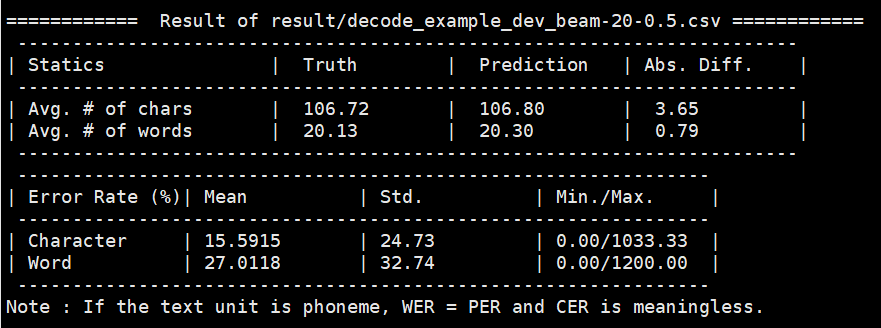
#### 4.2.2 SE+ASR

##### 验证集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_dev\_output.csv

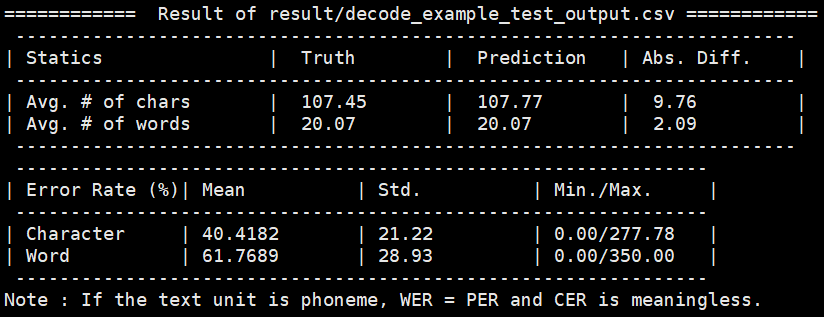


decode\_example\_dev\_beam-20-0.5.csv：

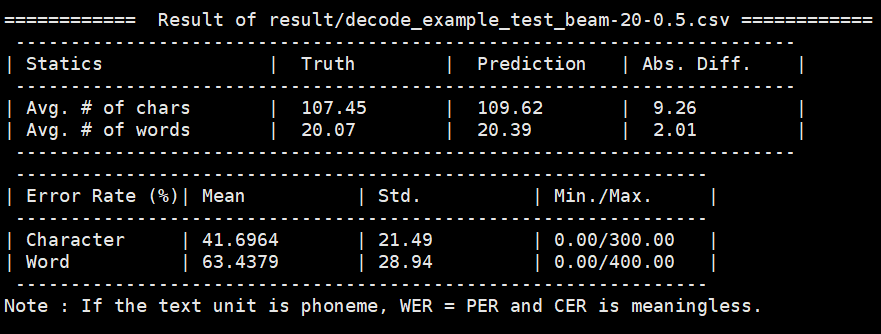


##### 测试集结果：

CUDA\_VISIBLE\_DEVICES=2 python3 eval.py --file result/decode\_example\_test\_output.csv



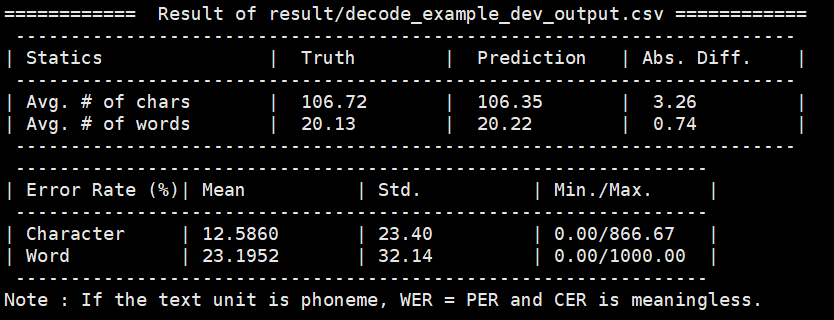
decode\_example\_test\_beam-20-0.5.csv：



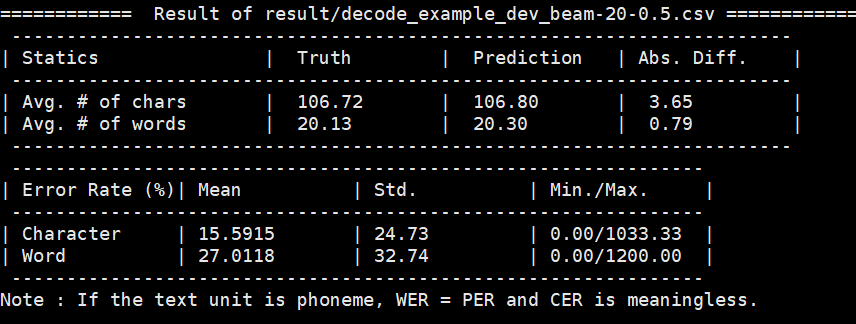
#### 4.2.3 SE\_Fbank+ASR

##### 验证集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_dev\_output.csv

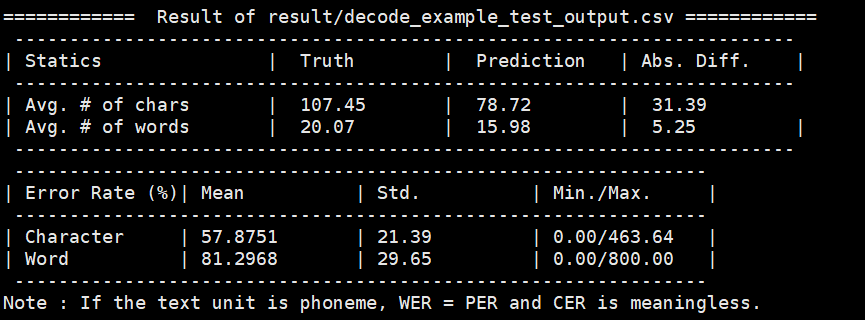


decode\_example\_dev\_beam-20-0.5.csv

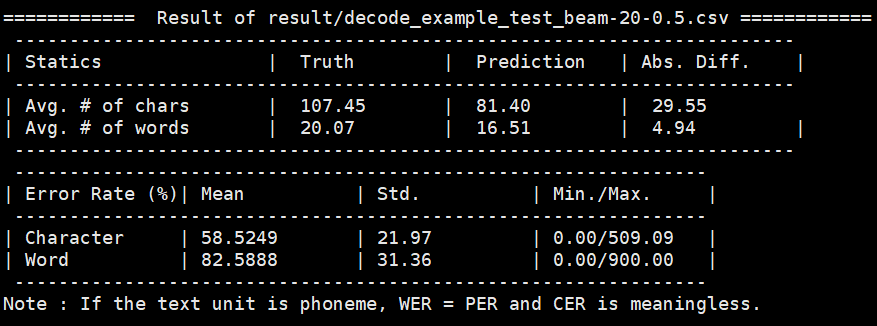


##### 测试集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_test\_output.csv



decode\_example\_test\_beam-20-0.5.csv：

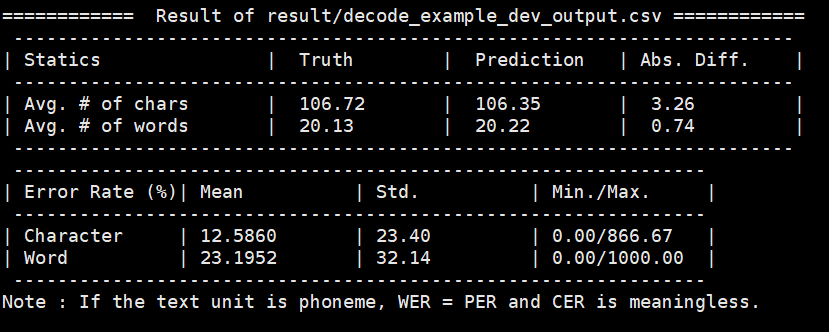


### 4.3思路三的实验结果

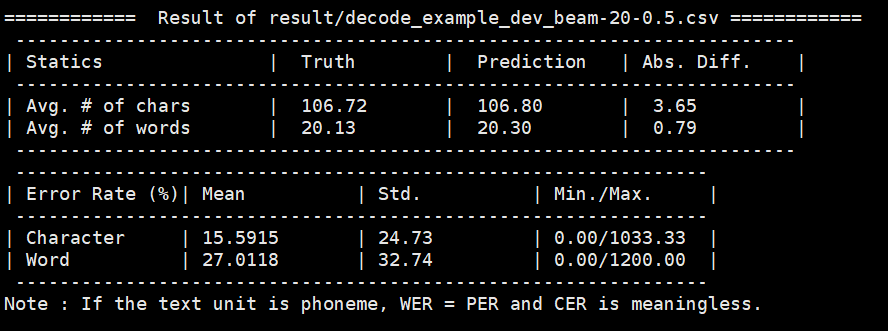
#### 4.3.1 SE\_Fbank+ASR

##### 验证集结果：

CUDA\_VISIBLE\_DEVICES=0 python3 eval.py --file result/decode\_example\_dev\_output.csv

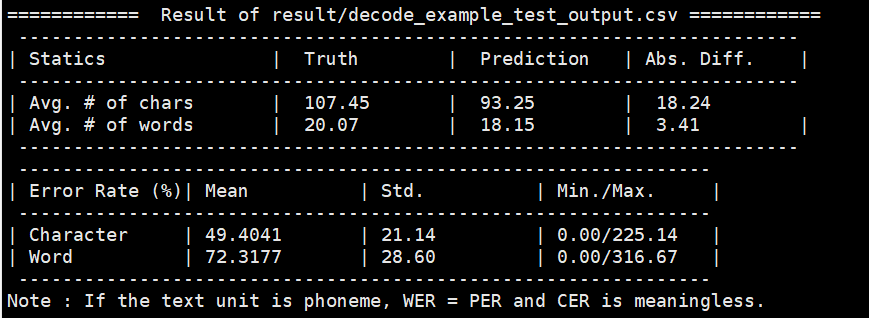


decode\_example\_dev\_beam-20-0.5.csv

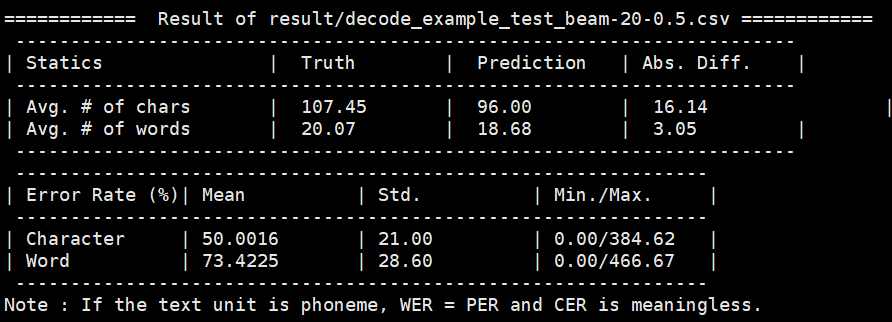


##### 测试集结果：

CUDA\_VISIBLE\_DEVICES=3 python3 eval.py --file result/decode\_example\_test\_output.csv



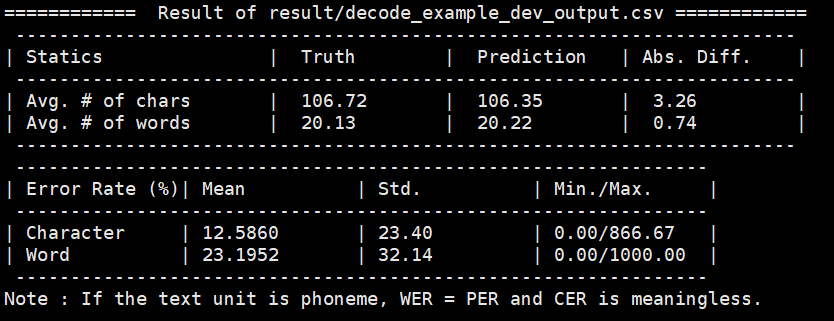
decode\_example\_test\_beam-20-0.5.csv：



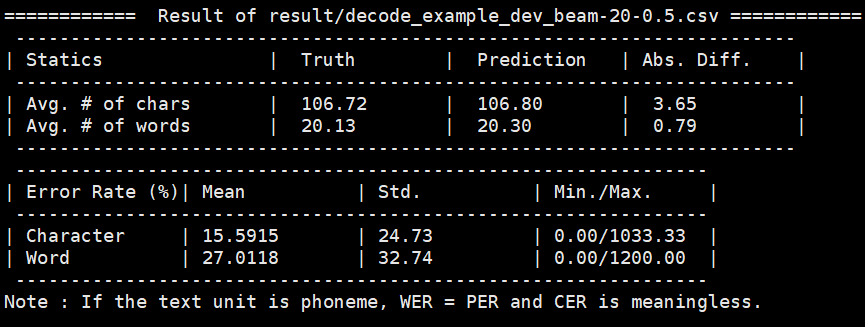
#### 4.3.2 ASR\_clean

##### 验证集结果：

CUDA\_VISIBLE\_DEVICES=0 python3 eval.py --file result/decode\_example\_dev\_output.csv

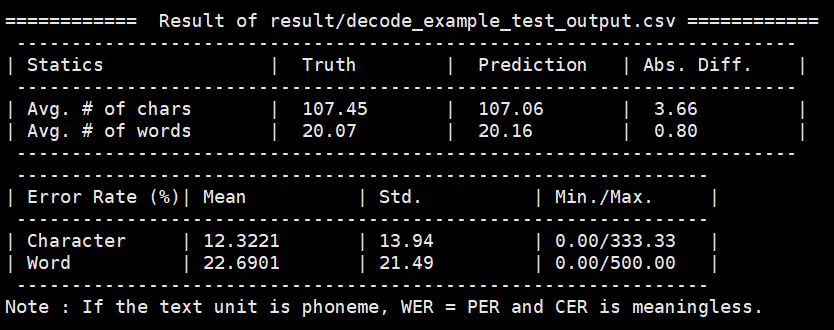


decode\_example\_dev\_beam-20-0.5.csv

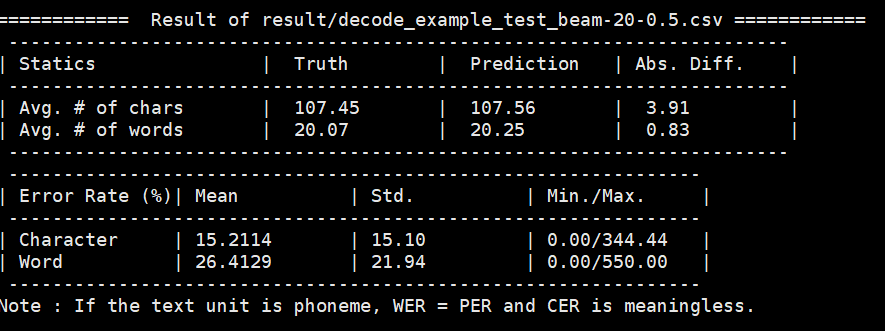


##### 测试集结果：

CUDA\_VISIBLE\_DEVICES=0 python3 eval.py --file result/decode\_example\_test\_output.csv



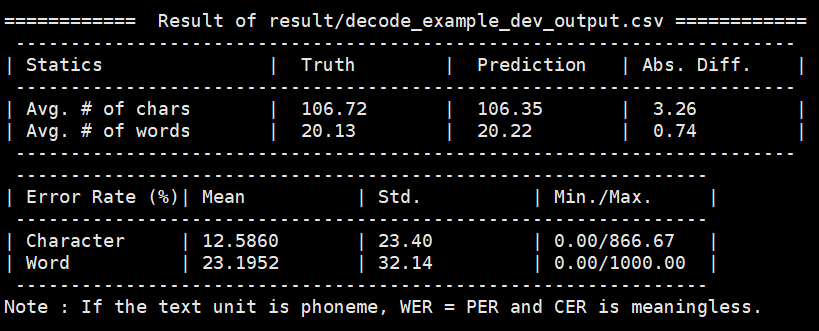
decode\_example\_test\_beam-20-0.5.csv：



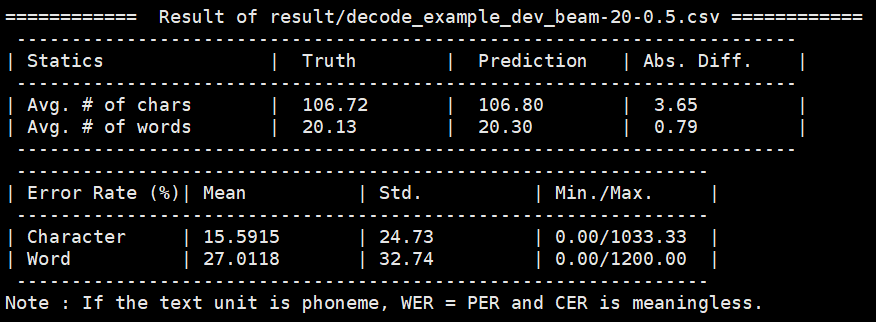
#### 4.3.3 ASR\_noisy

##### 验证集结果：

CUDA\_VISIBLE\_DEVICES=0 python3 eval.py --file result/decode\_example\_dev\_output.csv

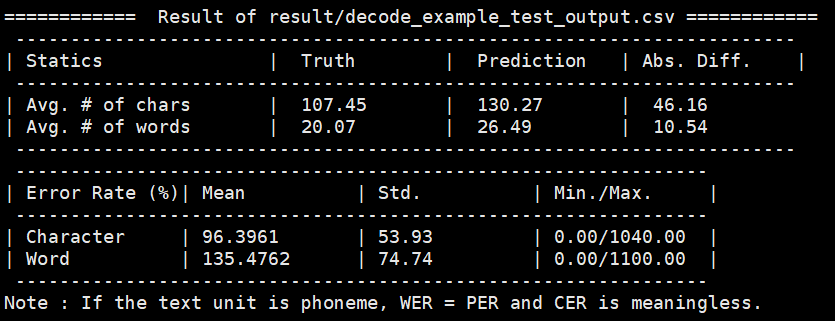


decode\_example\_dev\_beam-20-0.5.csv

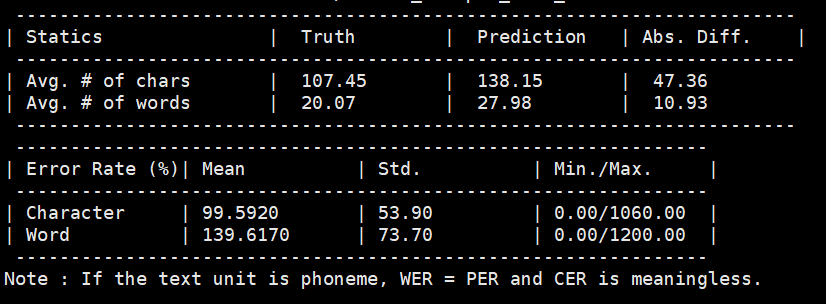


##### 测试集结果：

CUDA\_VISIBLE\_DEVICES=0 python3 eval.py --file result/decode\_example\_test\_output.csv



decode\_example\_test\_beam-20-0.5.csv：



## 5、Conclusions

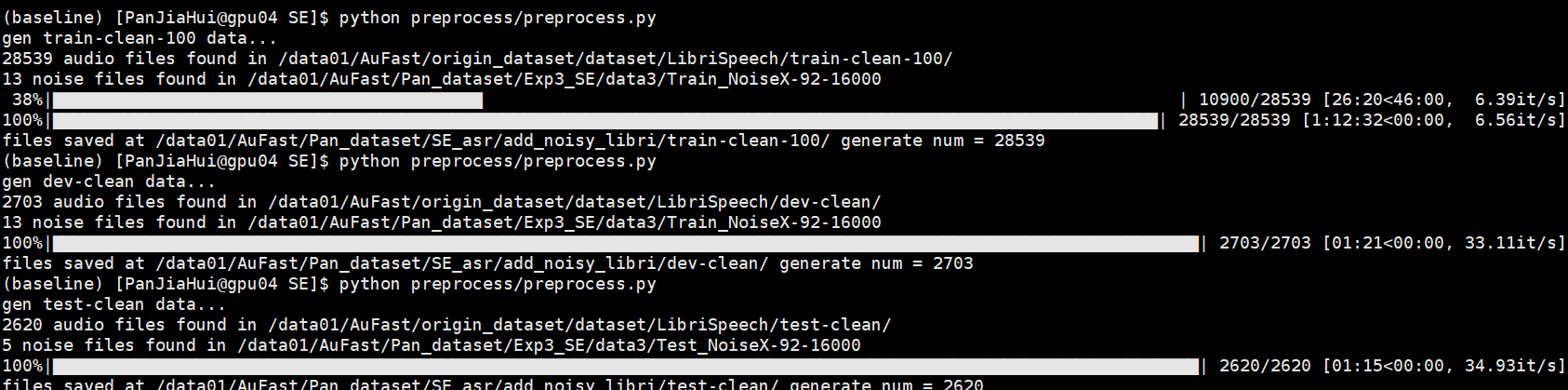
## 实验二步骤：

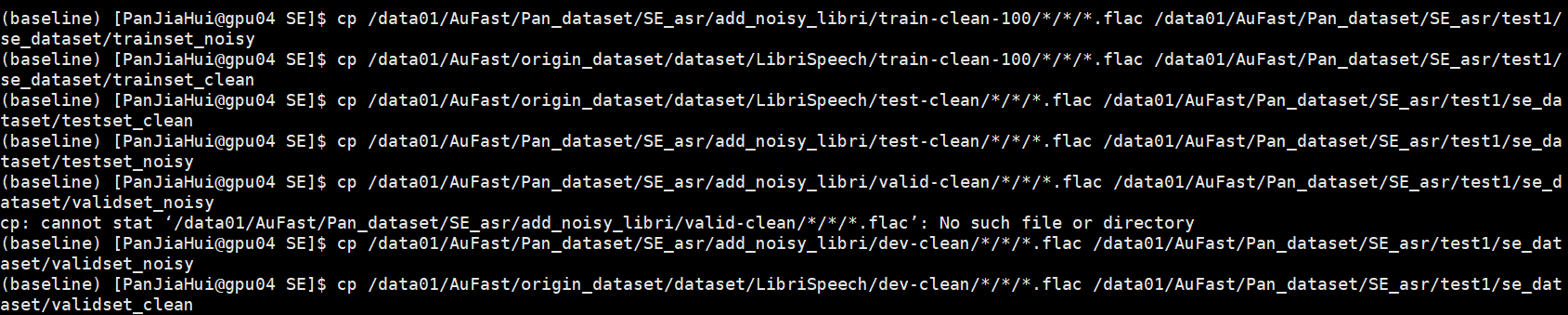
### 问题：se\_fbank用的训练集和验证集，与测试集的信噪比，噪声类型一样

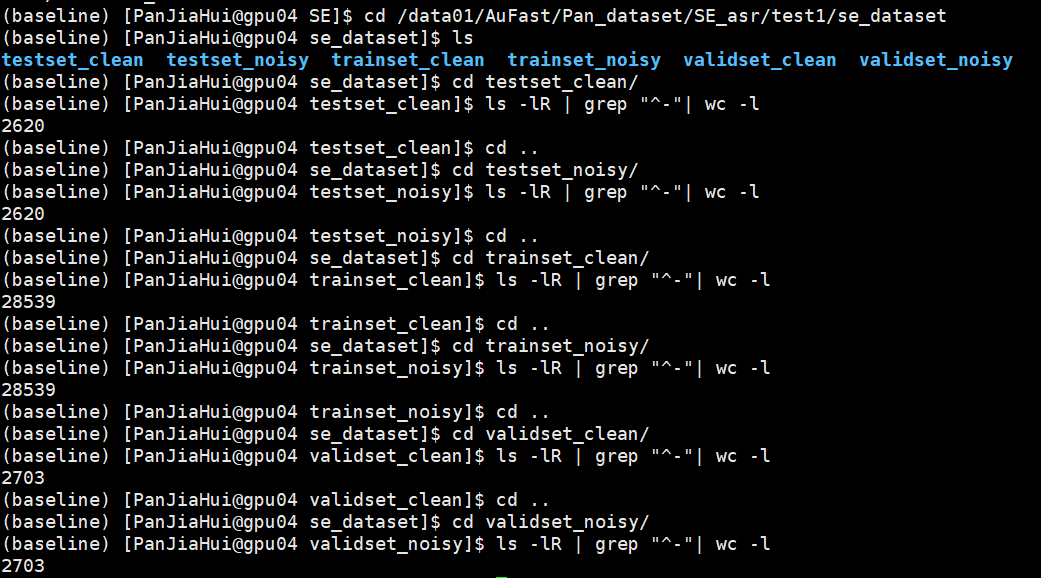
需要重新生成训练集和验证集

需要修改se\_fbank实验，se实验，重新训练，重新进行测试

#### 1.（完成）生成训练集和验证集



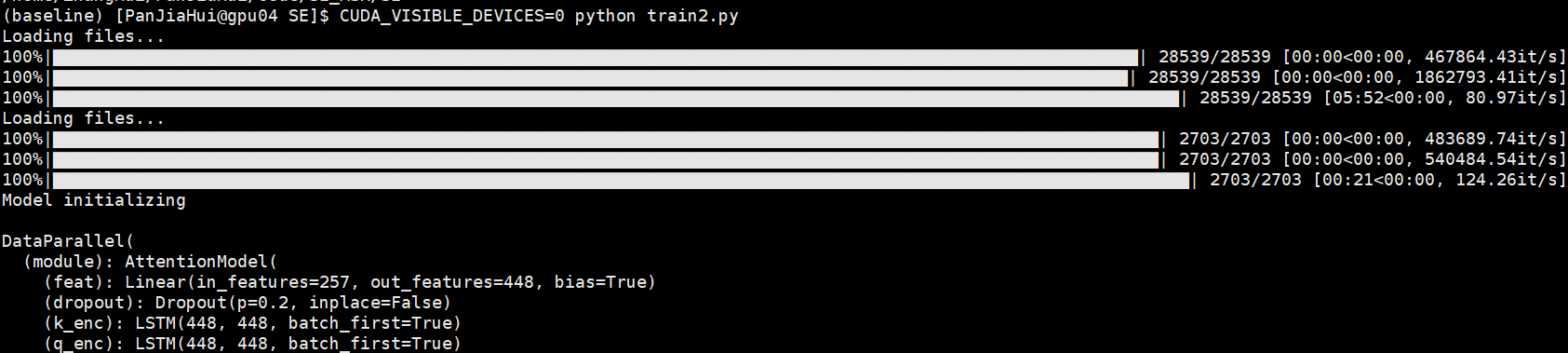


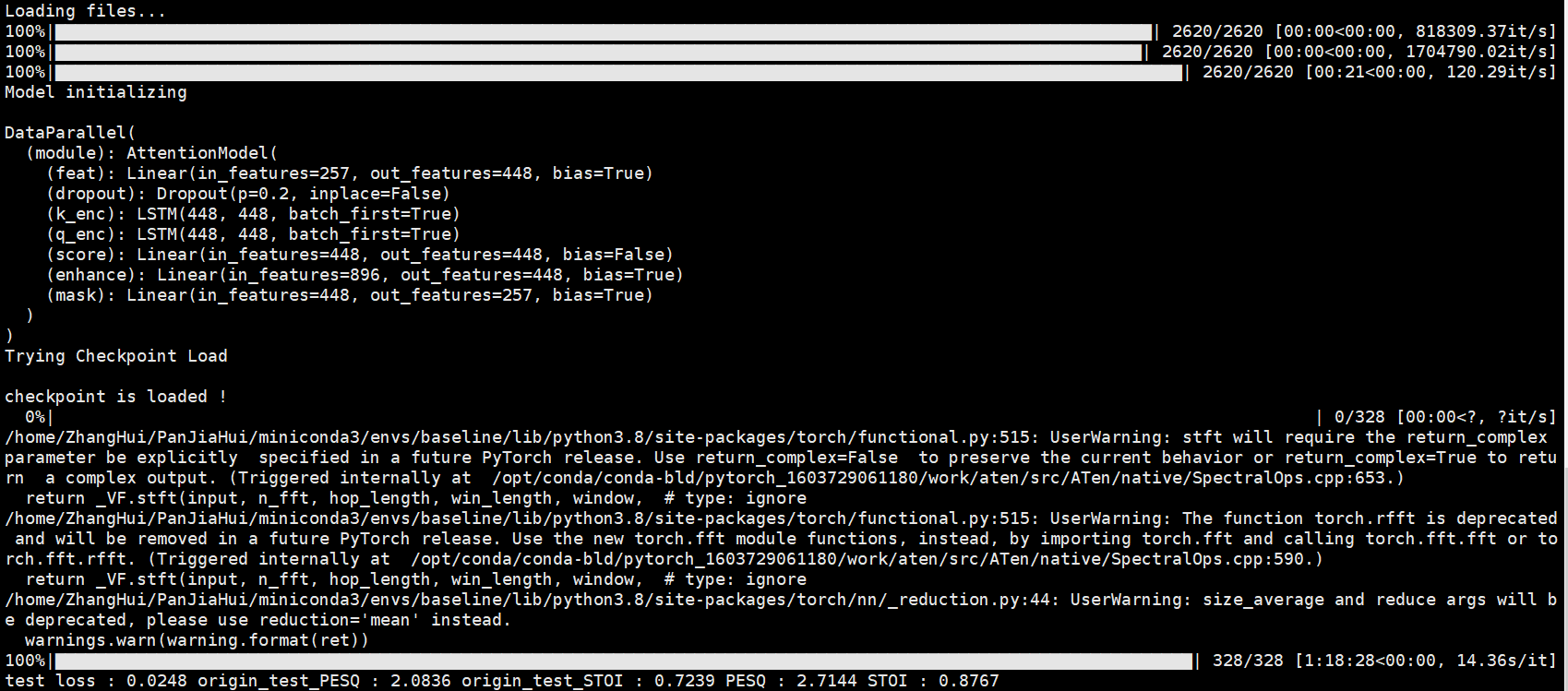


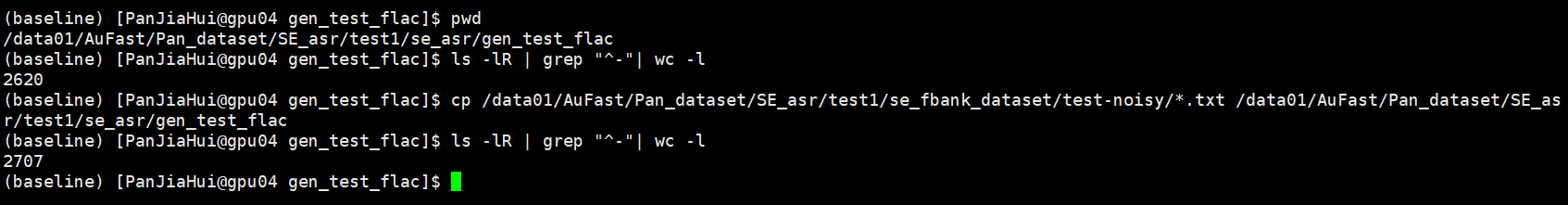
#### 2.（完成）修改se实验，重新训练，测试，得到生成的mix\_test

数据：/data01/AuFast/Pan\_dataset/SE\_asr/test1/se\_dataset

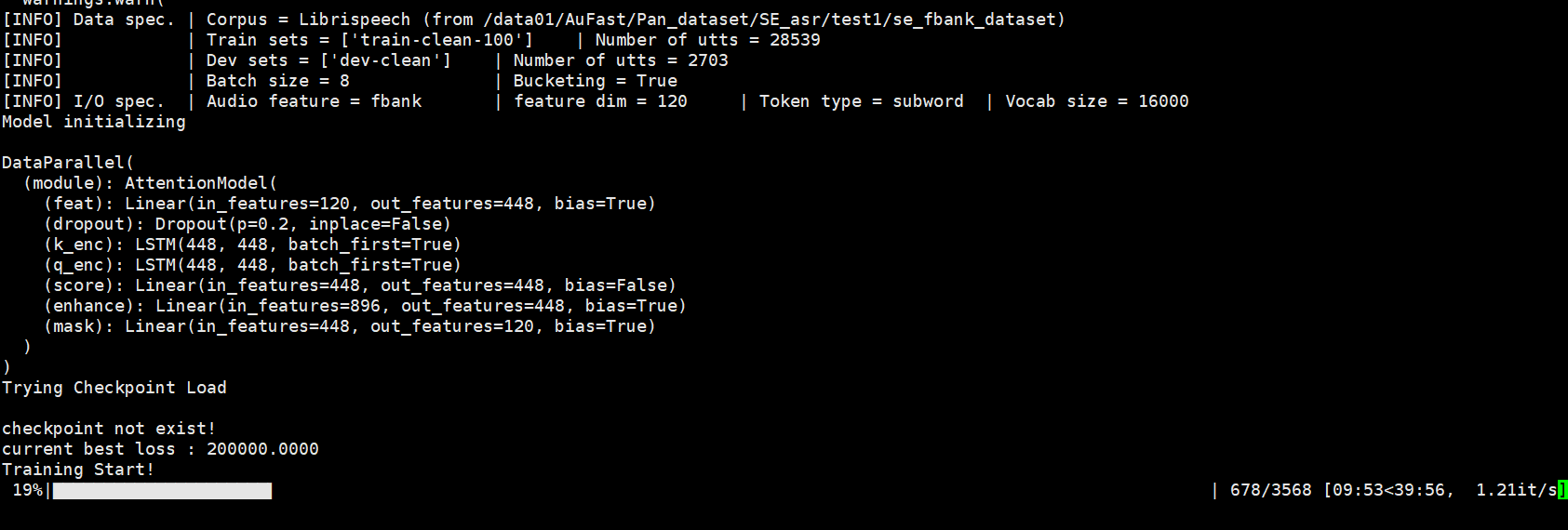
screen -r SE:CUDA\_VISIBLE\_DEVICES=0 python train2.py





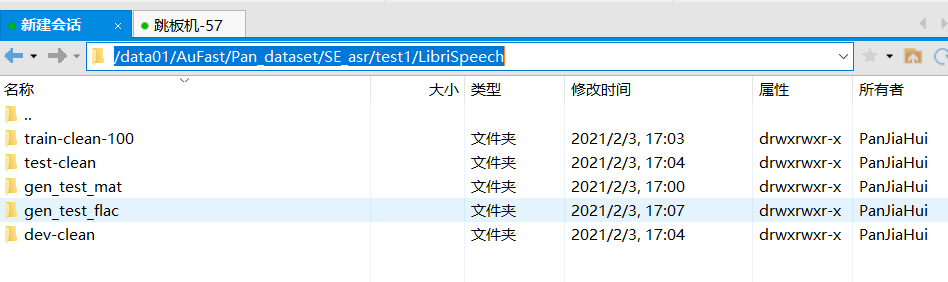


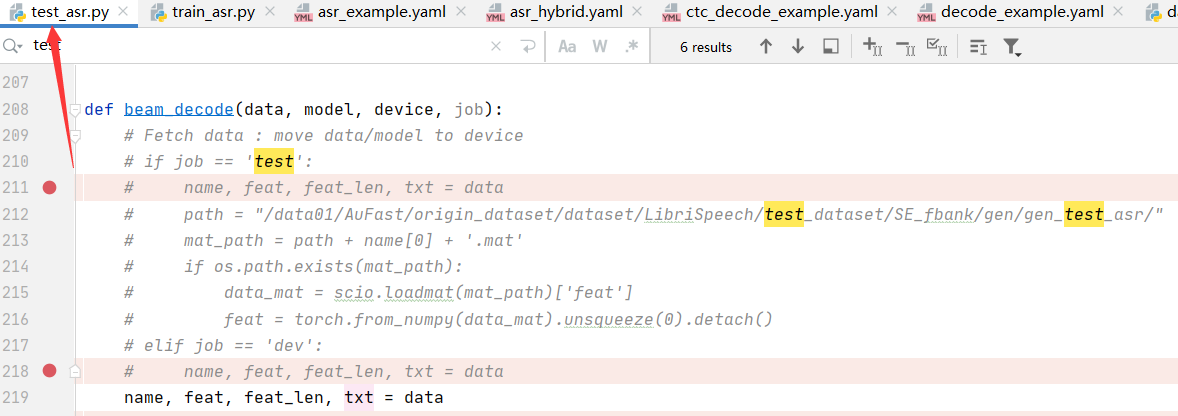
#### 3.（进行中）修改se\_fbank实验，重新训练，测试，得到保存的.mat



#### 4.（完成）Se+ASR(注意ASR处有修改，记得复原)

/data01/AuFast/Pan\_dataset/SE\_asr/test1/LibriSpeech





#### 5.（完成）SE\_Fbank+ASR(注意修改test\_asr)

#### 6.（完成）ASR\_clean

#### 实验二结束

# ASR论文：ADVERSARIAL TRAINING OF END-TO-END SPEECH RECOGNITION USING A CRITICIZING LANGUAGE MODEL

在这篇论文中对于端到端的语音识别采用了一种批判语言模型（CLM）来进行对抗训练。在这样，CLM和自动语音识别（ASR）模型可以相互挑战和互相学习，以提高性能。 由于CLM仅将文本作为输入，因此可以在端到端培训中以这种方式利用大量不成对的文本数据。 此外，可以使用任何基于深度学习的语言建模框架将AT应用于任何端到端ASR模型，并与任何现有的端到端解码方法兼容。 带有示例性实验设置的初步结果表明，所提出的方法能够在不同情况下从辅助文本数据中高效地获得一致的改进。

WER（词错率）

CER（字错率）

Mean（平均值）

Std（标准差）

# 验证实验1：

1. SE\_fbank中，使用使用libri训练集，验证集，训练网络，然后使用经过语音增强后的flac文件进行测试，保存mat文件。

路径：

/data01/AuFast/Pan\_dataset/SE\_asr/test/se\_fbank/libri

训练集验证集，信噪比0，-5

测试集，信噪比0，-5，5

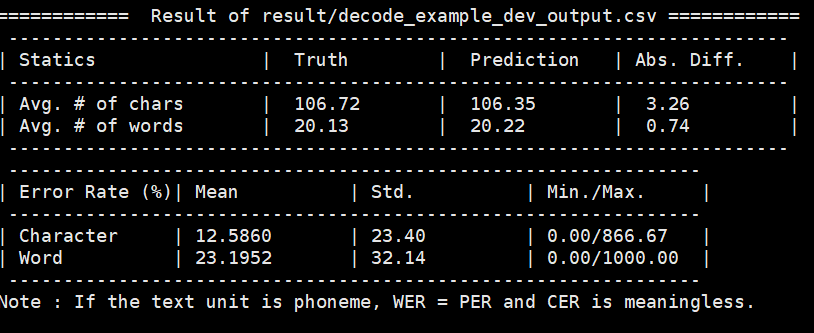
详情csv：/data01/AuFast/Pan\_dataset/SE\_asr/add\_noisy\_libri

**se\_fbank\_test.pt**

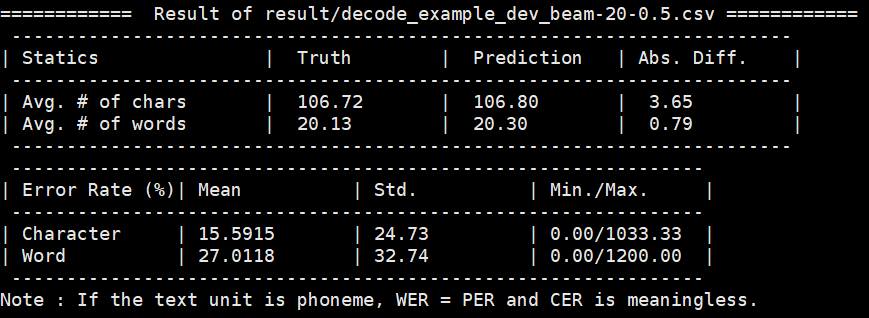
保存mat文件：/data01/AuFast/Pan\_dataset/SE\_asr/test/se\_fbank/libri/gen\_test\_mat/

1. 把mat文件，送入ASR中，检查实验结果

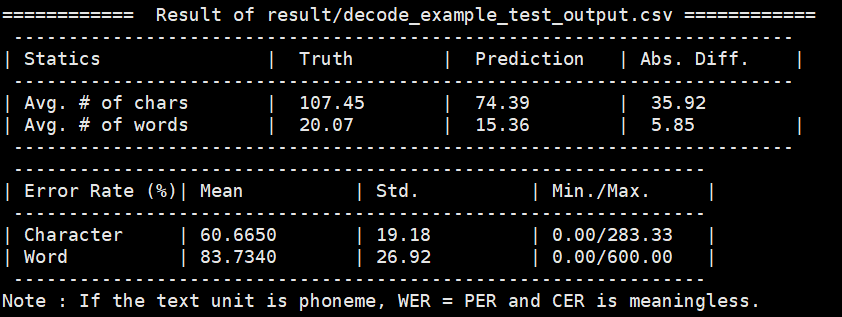
python3 eval.py --file result/decode\_example\_dev\_output.csv



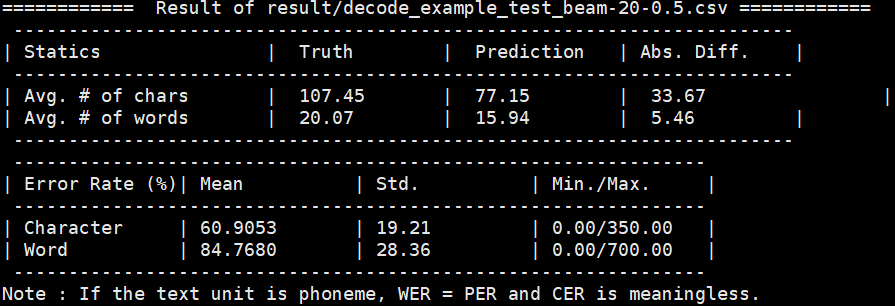
decode\_example\_dev\_beam-20-0.5.csv



python3 eval.py --file result/decode\_example\_test\_output.csv



decode\_example\_test\_beam-20-0.5.csv



# 验证实验2：

## 1、Se:wsj0+timit数据集，训练，用librispeech测试，生成flac（训练、测试、验证）,作为带噪语音

### 数据集：

训练集：（14494）：TIMITS（TIMIT-TRAIN）+ wsj0（si\_tr\_s）；信噪比：-5，0

验证集：（1857）：wsj0（si\_et\_05+ si\_dt\_05）；信噪比：-5，0

/data01/AuFast/Pan\_dataset/SE\_asr/finaltest

噪声类型以及信噪比：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 数据集 | 信噪比 | 噪声类型 |
| 训练集 | TIMITS（TIMIT-TRAIN）+ wsj0（si\_tr\_s）（14494） | -5，0 | volvo、machinegun、m109、leopard、hfchannel、factory2、factory1、f16、destroyerops、destroyerengine、buccaneer2、buccaneer1、babble |
| 验证集 | wsj0（si\_et\_05+ si\_dt\_05）（1857） |
| 测试集 | Libri:train-clean-100（29124） | -5，0，5 |
| Libri:dev-clean（2800） |
| Libri:test-clean（2707） | machinegun、leopard、factory1、destroyerops、buccaneer1 |

## 2、Se\_fbank: wsj0+timit数据集，训练，用librispeech测试，生成mat（训练、测试、验证）

## 3、Asr:librispeech生成的mat数据进行训练，测试，看结果