siyangbing / masr

mas... 分支 1 标签 1 文件 Web IDE 克隆/下载

libai update docker support 9b1b9e2 21 次提交

- docs
- examples
- images
- models
- .gitignore
- LICENSE.996
- README.md
- beamdecode.py
- data.py
- decoder.py

简介

暂无描述

发行版

暂无发行版

贡献者 全部

近期动态

feature.py

requirements.txt

train.py

README.md

MASR 中文语音识别

MASR是一个基于端到端的深度神经网络的开箱即用的中文普通话语音识别工具。

最新更新：现已支持docker！告别繁琐安装过程！参见使用docker

MASR的特点是：

- 开箱即用

  只需要不到5分钟，你就可以运行本项目并完成第一次中文语音识别

- 使用与训练分离

  直接使用该项目进行语音识别，识别过程不需要GPU。

- 识别率高！

  加入语言模型后的识别率完爆同类项目。

  不加语言模型的效果也是Github个人项目中最好的。

对比其他中文语音识别项目

直接说结论：

1. MASR提供的预训练模型的识别效果是个人开源项目中最好的
2. 和企业级开源项目相比，MASR使用起来更容易

详细的对比参见比较。

原理

MASR使用的是门控卷积神经网络（Gated Convolutional Network），网络结构类似于Facebook在2016年提出的Wav2letter。但是使用的激活函数不是`ReLU`或者是`HardTanh`，而是`GLU`（门控线性单元）。因此称作门控卷积网络。根据我的实验，使用`GLU`的收敛速度比`HardTanh`要快。如果你想要研究卷积网络用于语音识别的效果，这个项目可以作为一个参考。

以下用字错误率CER来衡量模型的表现，CER = 编辑距离 / 句子长度，越低越好

大致可以理解为 1 - CER 就是识别准确率。

模型使用AISHELL-1数据集训练，共150小时的录音，覆盖了4000多个汉字。工业界使用的语音识别系统通常使用至少10倍于本项目的录音数据来训练，同时使用特定场景的语料来训练语言模型，所以，不要期待本项目可以和工业界的识别效果媲美。这对于Github上任何个人项目来说都不现实，除非有更先进的技术诞生。

什么叫特定场景的语料训练的语言模型？比如你使用游戏中的语音识别，它更倾向于将你的话识别成你在玩游戏时可能说的话，比如「貂蝉被蓝打死了」。而在其他场景下，「貂蝉被蓝打死了」根本就不是一句通顺的话。不信你和一个只读过三国演义没玩过王者荣耀的人说「貂蝉被蓝打死了」，你确定*ta*不会反问你：「啥？貂蝉被谁打死了？*lan*是谁？」

在单卡GTX 1080Ti上，模型每迭代一个epoch大约需要20分钟。（实验室的CUDA版本较低，不排除更新CUDA版本后会快一些的可能。）

上图为验证集的CER随epoch的训练曲线。可以看到，目前验证集CER已经下降到11%。

图中没有显示测试集的表现。测试集的CER稍高一些，在14%。

通过外接语言模型可以将测试集的CER降低到8%。

项目目前提供的预训练模型训练了大约是100个epoch时候的，已经接近最好了。

MASR提供可以直接使用的预训练模型，如果你想自己训练，参见：训练

5分钟上手

使用语言模型会稍微麻烦一些，我们先来一次不使用语言模型的识别。但至少让你看到这个项目可以work，对不对？然后我们再一步步加上语言模型。

1. 克隆本项目到本地。

```
git clone https://github.com/libai3/masr.git
```

2. 从这里（提取码：xhks）下载预训练模型和测试音频文件，并将它们拷贝到对应位置。

```
mkdir masr/pretrained
cp ~/Downloads/gated-conv.pth masr/pretrained/
cp ~/Downloads/test.wav masr/
cd masr
```

3. 打开测试文件，听一下，说的是：「你好，很高兴认识你」。

4. 安装依赖。

```
pip install -r requirements.txt
```

5. 运行示例

```
python examples/demo-recognize.py
```

   你将看到识别结果。

   什么？效果不好？别走！看这里：使用语言模型提升识别率

识别自己的语音

别急，先别急着喷识别率，先试试识别一下自己的语音。

识别自己的语音需要额外安装一个依赖：pyaudio，它是用来进行录音的。

在mac上安装pyaudio的方法如下：

```
brew install portaudio
pip install pyaudio
```

第一步必须先执行哦。

其它系统的可以参考pyaudio官网把它装上（很方便的），然后执行以下命令即可。

```
python examples/demo-record-recognize.py
```

请在看到提示「录音中」后开始说话，你有5秒钟的说话时间（可以自己在源码中更改）。

使用语言模型提升识别率

我能理解，当你看到「你好很高兴认识你」被识别成「利好很高性任实米」时，一定会感觉很失望。甚至觉得我在骗你：这怎么可能是Github上效果最好的个人项目？

也许你看到别人的首页上展示的识别效果都是全对的，感觉很厉害。其实它们展示的是测试集上部分样本的效果，测试集跟训练集是同分布的，效果不会差。甚至有一些项目展示的是训练集上的效果，这就像高考考做过的题一样，毫无意义。本项目在测试集上的识别率高达86%（不含语言模型的情况下），拿出来一点也不虚。

但你可能会说：「刚才那句话哪有86%的识别率，就识别对了3个字好吗？」哎，别急，你看看，虽然字只对了3个，但错的那些字是不是音也很接近呢？

说了那么多，并不是要告诉你，只能做成这样了，别再奢求更好的结果了。不！我们可以把「你好很高兴认识你」识别到全对，还能把测试集上的识别率提升到92甚至更高。而我们需要的是一个好的语言模型。

什么是语言模型

如你所见，「利好很高性任实米」根本就不是一句通顺的话，但是我们的神经网络并没有意识到这一点。为什么呢？也许你不相信，因为从来没有人对它说过「你好」这句话。训练集中没有「你好」这句话，也没有「很高兴认识你」（训练集只有150个小时）。它不知道这些话是合理的句子（至少比「利好很高性任实米」更合理）。

而语言模型是通过外部语料库训练得到的，这些语料库包含大量的句子（如果读出来的话

远超150h的录音时长），语言模型知道「你好很高兴认识你」比「利好很高性任实米」更像一个句子。换句话说，神经网络的第一个字输出了「利」，但是「你」的概率也并不低，而语言模型可以纠正它的错误，告诉它，真正该输出的是「你」。

下面我们来一步步加入语言模型，完成后你将会得到一个可以说是惊喜的识别效果。别担心，不需要写任何代码。

添加语言模型

添加语言模型需要一些额外的依赖，你可以自行安装或使用docker。

如果你了解docker，或者自行安装时或安装完后运行masr时遇到错误，则推荐使用docker。

自行安装

1. 在你尝试添加语言模型之前，请确认你已经安装了`pyaudio`，参见识别自己的语音。
2. 同时，你还需要安装Flask，这很简单，`pip install flask`即可。我们需要它来启动识别的web服务。
3. 好了，让我们给本项目加入一个来自百度的语言模型。

   一个现成的可以使用的语言模型来自百度，你需要下载它。（大概2个多G，但应该很快）

   这一步很简单，因为从百度下载很快。

   下载完成后，执行

   ```
   cd masr
   mkdir lm/
   cp ~/Downloads/zh_giga.no_cna_cmn.prune01244.klm lm/
   ```

   将语言模型拷贝到`masr/lm`目录下。
4. 下面是最麻烦的一步了，做好准备。

   我们需要安装这个依赖：ctcdecode，这是一个高效的beam search解码器。

按照它的README，安装它本来很简单，你需要执行：

```
git clone --recursive https://github.com/parlance/ctcdecode.git
cd ctcdecode
pip install .
```

但可能会遇到报错，原因在于它在安装过程中会去下载两个文件，而这两个文件位于Google的服务器上。你可能需要魔法才能访问。而你的命令行可能不会自动使用魔法。

以下是`build.py`中下载文件部分的代码

```
# Download/Extract openfst, boost
download_extract('https://sites.google.com/site/openfst/home/openfst-
down/openfst-1.6.7.tar.gz',
                 'third_party/openfst-1.6.7.tar.gz')
download_extract('https://dl.bintray.com/boostorg/release/1.67.0/source/boos

                 'third_party/boost_1_67_0.tar.gz')
```

你需要自行下载这两个文件，并把它们解压到`third_party`目录下，然后注释掉这两行。再次执行安装命令`pip install .`，即可成功安装。

5. 好了，恭喜你，已经完成了所有依赖的安装，现在，启动服务器

```
python examples/demo-server.py
```

然后，请将`examples/demo-client.py`中的服务器地址的ip部分改成你的服务器ip，如果你都是在本机上进行的，则不需要修改，使用默认的`localhost`即可。

使用docker

docker使用起来就像虚拟机一样，同时有着原生的性能。使用docker来运行masr非常方便。

```
docker pull libai3/masr-env:latest

cd masr
docker run -v $(pwd):/workspace/masr -p 5000:5000 libai3/masr-env:latest
```

感受喜悦

好了，执行

```
python examples/demo-client.py
```

在看到「录音中」的提示后，开始说话。

如果想不到说什么，不妨念「举头望明月，低头思故乡」试试。

接下来，就是见证奇迹的时刻。

训练

你可以使用自己的数据集来训练模型。你的数据集需要包含至少以下3个文件：

- train.index
- dev.index
- labels.json

train.index和dev.index为索引文件，表示音频文件和标注的对应关系，应具有如下的简单格式：

```
/path/to/audio/file0.wav,我爱你
/path/to/audio/file1.wav,你爱我吗
...
```

labels.json是字典文件，应包含数据集标注中出现过的所有字符，表示为一个json数组。其中第一个字符必须是无效字符（可任意指定，不和其他字符重复就行），预留给CTC作为blank label。

```
[
```

```
    '_', // 第一个字符表示CTC空字符，可以随便设置，但不要和其他字符重复。
    '小',
    '时',
    '不',
    '识',
    '月',
    ...
]
```

准备好数据集以后，就可以开始训练了。examples/train.py目前有一些问题，请暂时使用项目根目录的train.py来训练模型，它提供了训练用的函数，请注意修改预训练模型的保存路径（默认为pretrained/model_{epoch}.pth）。

Issue区规范

如果你在使用时遇到问题或者有什么想法，可以在issue区提出来。

大家都是聪明的人，想必都明白准确清晰的提问更容易得到热心的帮助；同理，低质量的Issue会不受理睬或被close掉。