CSLT TECHNICAL REPORT-20160016 [Friday 6th January, 2017]

Kaldi 中多任务联合训练的实现方法

Hang Luo[1]
, Zhiyuan Tang[1]
and Dong Wang[1*]

*Correspondence:
wangd99@mails.tsinghua.edu.cn,
[1]Center for Speech and Language Technologies, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China
Full list of author information is available at the end of the article

Abstract

近年来深度学习技术在语音信号处理领域得到广泛应用，然而目前的神经网络模型常常将相互关联的语音任务分离开来独立对待，例如自动语音识别（Automatic Speech Recognition，ASR）[1] 和说话人识别（Speaker Recognition）[2]。这种单一任务的处理模式，显然不符合人类大脑的工作方式，也不利于任务之间知识的共享和交互。基于此，[3,4] 提出了多任务联合训练（Multi-task joint training）的方法，此方法通过训练一个统一模型来完成多个任务的学习。它的优势在于，训练过程中各个子任务模块的信息实时交互，从而促进各部分子系统的性能。本文基于Kaldi [?]，以 [4]中的自动语音识别和语言识别这两个任务为例，介绍了二者联合训练的实现方法。文中所述的相关步骤或过程可拓展到其他任务，比如语音识别与说话人识别 [3]、说话人识别与语言识别，甚至是两个以上的多任务协同学习。另外，本文着重描述joint training的关键步骤，对基于DNN-HMM结构的语音识别基本流程（可参见Kaldi WSJ Recipe）不再赘述。

Keywords: 多任务联合训练; 语音识别; 语言识别

1 简介

图1是语音识别和语言识别联合训练时的总体框架，可以看到语音识别部分的声学模型和语言识别模型的特征输入是相同的，二者在训练时实时交互信息。为了实现以上系统，需要在WSJ nnet3 recipe上做出3个方面的修改，分别对应图1中数据准备、模型训练、解码三个步骤。本文的基本实验配置与 [4]相同，即采用中英两种语言的数据集：Thchs30和Aurora4，且已完成两个数据集的基本语音识别系统（对应 [4] 中的baseline1实验）。接下来将详细介绍每个部分的修改。

2 数据准备

对于有监督学习来说，训练数据无非是样本的输入数据及其对应的标签。对于语音识别和语言识别任务来说，它们所需的数据又有些区别。

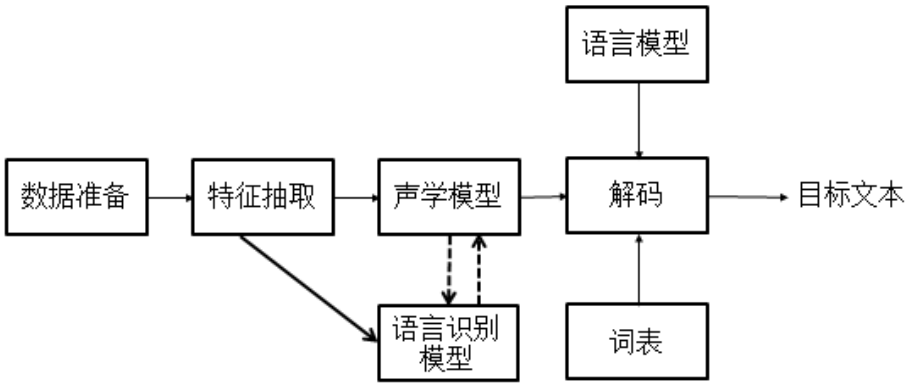

Figure 1: 语音识别和语言识别的联合训练流程

2.1 声学模型相关数据

语音识别系统所需的基本数据包括两部分。一部分是特征相关的数据（放于文件夹data/），主要包括作为系统**输入**的语音特征（由音频文件提取出，如MFCC、Fbank），及其对应的转录文本、说话人信息等。另一部分是发音相关的辅助数据（放于文件夹data/dict/），主要包括音素、词典相关的说明文件。Thchs30和Aurora4在Kaldi中有标准的Recipe，可以据此获取各自所需的基本数据。为了能够进行联合训练，我们需要将两种语言的基本数据集进行融合。

其中，data/中的文件分布如下：

cmnv.scp	feats.scp	spk2utt	text	utt2spk	wav.scp

将两个不同的data/融合，可使用utils/combine_data.sh，方法如下：

utils/combine_data.sh	dest-data-dir	src-data-dir1	src-data-dir2

data/dict/中的文件分布如下：

extra_questions.txt	lexicon.txt	nonsilence_phones.txt
optional_silence.txt	silence_phones.txt	

将两个不同的data/dict/融合，对其直接合并且去掉重复项即可。本例中，静音在英文lexicon.txt中标注为!SIL，在中文lexicon.txt中为SIL。融合时统一保留为!SIL。

data/和data/dict/分别融合完毕后，按照WSJ Recipe训练出基本的GMM-HMM 系统，并用GMM标记语音特征的标签（作为声学模型的**输出**），到此，上述联合训练中声学模型部分所需数据（输入、输出）已准备完毕。

2.2 语言识别相关数据

语言识别是个常规的分类问题，它的输入与声学模型部分相同，已准备完毕，它的输出标签，即每帧语音数据所对应的语言类型，需要人工标记。如果不考虑静音的影响，可将每帧的语言标识标注为0或1，分别代表英文或中文。标记时，可用feat-to-len命令得到每句话的帧数，之后根据当前句子的语言类别对其进行标注，相应文件的格式与声学模型部分的标记文件（由GMM得出，放于exp/tri4b_ali/）一致，如下（行首的字符串是句子的ID，其后是每帧的标签）：

feats-to-len使用方法如下：

feats-to-len	in-rspecifier	out-wspecifier

语言标记文件的格式如下。其中第一列为每个句子对应的id，第二列至最后一列为一个句子中每帧对应的语言标记id。下面例子中第一行对应英文数据库中id为011c02011的句子，后一句对应中文数据库中id为C8_749的句子。

011c02011	0	0	...	0
...				
C8_749	1	1	...	1

到此，上述联合训练中语言识别部分所需数据（输入、输出）也已经齐全了。

3 模型训练

所有数据准备完毕后，开始进行模型的训练。模型训练分为单任务训练（作为baseline）和多任务联合训练，单任务训练中，语音识别和语言识别需要分别进行。

3.1 声学模型训练

单独训练的DNN声学模型仍可按照WSJ nnet3 recipe进行，相关过程不需要更改。

3.2 语言识别模型训练

因为语言识别模型的输入与声学模型相同，标记数据的格式也与GMM的标记格式一致，所以单独训练的语言识别模型可参照WSJ nnet3 recipe，在训练模型时进行细微修改即可进行。例如，在语言识别模型训练中，不像声学模型需

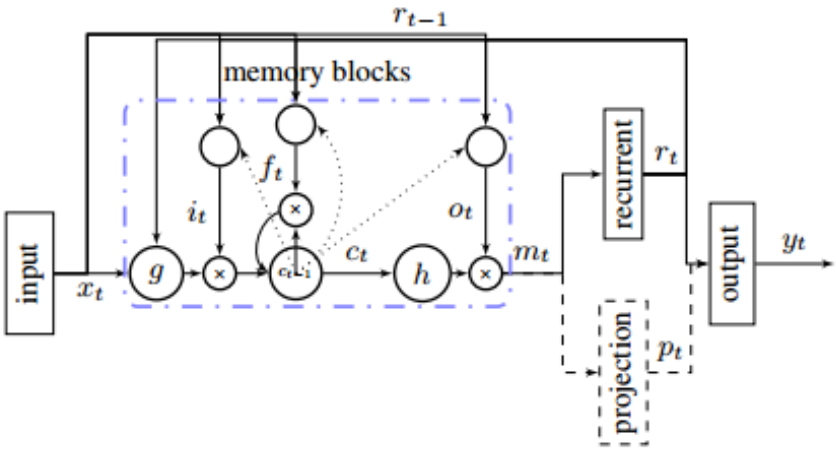

Figure 2: LSTM中单个记忆单元的结构

与GMM模型的后验概率对齐，且无需HMM状态转化信息，故在训练脚本中与之相关的部分均可删除。具体的实现脚本可参考：

/work3/luohang/joint_process/steps/nnet3/lstm/train_lan_pure.sh

3.3 多任务联合训练

在单任务的语音识别任务中，声学模型所用的神经网络仅对应一个输出，即各pdf的概率。语言识别模型加入后，联合模型便有了两个部分，各自有输出，且信息实时交互。此时Kaldi nnet3预定义的神经网络已不满足需求。Kaldi nnet3的初始化是通过读取配置文件（放在exp/nnet3/configs/）完成的，为了获得符合要求的网络结构，可以预先写好配置文件（实验中不再需要配置文件自动生成）。联合训练模型是基于LSTM实现的。在单任务的语音识别任务中，nnet3预定义的2层LSTM配置文件可参考该目录下的配置文件：

/work3/luohang/speech_lan/bilingual/exp/nnet3/lstm/configs

与nnet3预定义的LSTM配置文件相比，需要作出的修改如下：

(1)**信息交互**。信息交互需要模型的一部分作为信息的提供者，另一部分作为信息的接收者。LSTM有很多不同的组成部分，LSTM中一个记忆单元（memory block）的结构如图2所示 [5]：

这使得基于LSTM的联合训练可以有多种不同的交互方式。例如信息的提供者可以是recurrent信息r_t,也可以是non-recurrent信息p_t，或者是二者的组合。信息的接收者，可以是输入门i_t，输出门o_t，遗忘门f_t，或者非线性函数g(.)。以图3 为例，

将recurrent信息r_t和non-recurrentp_t信息，提供给另一个模型的非线性函数g(.)。修改后的配置文件如下：

```
/work3/luohang/speech_lan/bilingual/exp/nnet3/lstm_joint_g_s_2_p/configs/all.config
```

配置文件中将定义两个LSTM模型，其中以Lstm1开头的组件和声学模型相关，以lan_Lstm1开头的组件和语言识别模型相关。实现信息交互的改动如下：

声学模型中，给g门加入语言识别模型信息:

```
component-node name=Lstm1_g1 component=Lstm1_W_c-xr
input=Append(L0_lda, IfDefined(Offset(Lstm1_r_t, -1)),
IfDefined(Offset(lan_Lstm1_rp_t, -1)))
```

语言识别模型中，给g门加入声学模型信息:

```
component-node name=lan_Lstm1_g1 component=lan_Lstm1_W_c-xr
input=Append(L0_lda, IfDefined(Offset(lan_Lstm1_r_t, -1)),
IfDefined(Offset(Lstm1_rp_t, -1)))
```

(2)**多个输出**。

对于多个输出的更改，这个比较简单。结构中有两个LSTM子模型，每一个均有一个output-node，训练时将想要结果的任务的output-node的名称设定为output即可。假设我们想要输出声学模型的结果，将Lstm1中的output-node名称设定为output，对于语言识别模型lan_Lstm1的output-node设置为别名即可。

在我们的实验中设置如下，其中参数5代表label_delay，即将LSTM模型的输出用于预测5帧后的label。

声学模型中中output-node:

```
output-node name=output input=Offset(Final_log_softmax,5)
```

语音识别模型中output-node:

```
output-node name=output-plus input=Offset(Final_log_softmax,5)
```

如果想查看语言识别模型的结果，可使用nnet3-am-copy命令，将训练好的final.mdl转成raw格式后修改语言识别模型output-node名称，然后使用nnet3-compute 命令即可得到模型输出结果。

```
nnet3-am-copy –binary=false final.mdl final.raw
```

然后将语音识别模型的output-node名称设置为output,声学模型output-node改为别名

```
nnet3-compute final.raw scp:feats.scp ark:nnet_prediction.ark
```

noindent

对于此配置文件的完整版本，可参考：

/work3/luohang/speech_lan/bilingual/exp/nnet3/lstm_joint_g_s_2_p/configs/all.config

值得一提的是，Kaldi nnet3将训练集的输入输出事先写入特定格式的文件egs中（放于exp/nnet3/egs/），训练时直接调用，以节省时间开销，而egs的输出个数与神经网络的输出个数是一致的，因此，当网络中的输出个数发生改变时，egs也应该做相应处理。egs是通过nnet3-get-egs命令获取的，nnet3-get-egs目前只支持一个输出，应改写nnet3-get-egs为nnet3-get-egs-double-targets（两个输出），两个以上输出的更改方法类似。修改前后的文件可参考：

修改之前的nnet3-get-egs.cc
/work3/tzy/github/kaldi_master_20160126_interspeech16_cuda7.0/src/nnet3bin/nnet3-get-egs.cc
修改之后的nnet3-get-egs-double-targets.cc
/work3/tzy/github/kaldi_master_20160126_interspeech16_cuda7.0/src/nnet3bin/nnet3-get-egs-double-targets.cc

同时应改写原recipe中调用nnet3-get-egs的steps/nnet3/get_egs.sh以及上层脚本（传入两个任务的标签）。更改前后训练神经网络的脚本可参考：

修改前的train.sh
/work3/luohang/joint_process/steps/nnet3/lstm/train.sh
修改后的train_speech_lan.sh
/work3/luohang/joint_process/steps/nnet3/lstm/train_speech_lan.sh

另外，生成了多个任务的egs后，可以在每次跑不同网络时设置参数common_egs_dir，这样可以避免每次训练时重复生成egs。

至此，网络的结构配置好，与其相匹配的egs也做好了，联合模型的训练可以进行了。训练脚本仍有一些地方需要做细微的改动，改动的主要原因在于需要传入一些参数、调整部分不匹配处理步骤。

4 解码

Kaldi nnet3的解码工具默认处理网络中节点名称为output的输出，因此上述多任务神经网络（声学模型的输出命名为output）可以直接放入原解码脚本，不许做任何改动。

真正的多语言语音识别系统解码时同时共享声学模型和语言模型，如图5所示。本文中，中英混合语音识别系统共享一个声学模型，但是两种语言的语言模型（LM）还是分离的，如图4所示，因此，我们需要融合这两个语言模型，解码结果对应 [4] 中的baseline2 实验。

对于给定的两个语言模型，可使用ngram工具进行融合，其使用方式如下：

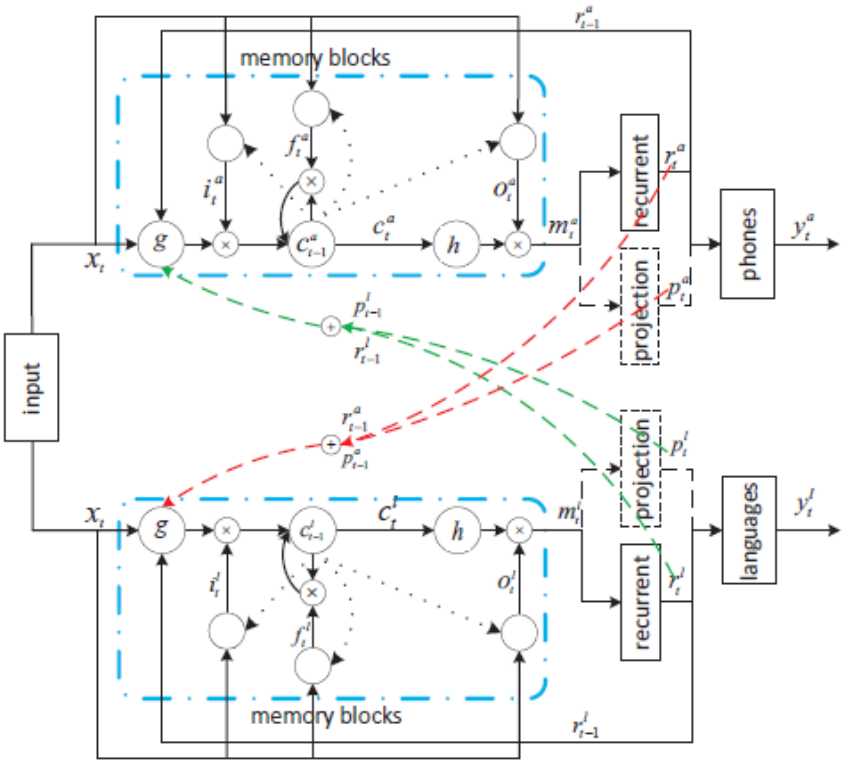

Figure 3: 联合训练的一种交互形式

ngram -lm $mainlm - order$num -mix-lm $mixlm - lambda - write - lm$mergelm

其中每个参数的含义为：

-lm	这个参数代表被融合的模型
-order	使用最大ngram长度
-mix-lm	代表加入的模型
-lambda	代表加入的模型在融合时的权重
-write-lm	最终的融合模型

当得到一个语言模型后，解码所需的WFST（命名为HCLG.fst）的生成方法与原recipe相同。

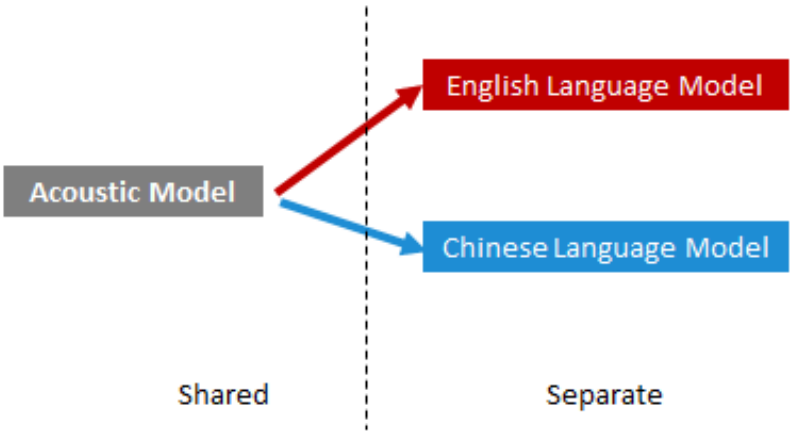

Figure 4: 分开的语言模型

5 总结

本文介绍了Kaldi中语音识别任务和语言识别任务联合训练的实现流程，对于想要复现及修改论文 [4]中模型的读者有一定的参考意义，同时读者也可参考此文档实现其他任务间的联合训练。

实验相关的所有代码放在/work3/luohang/joint_process和/work3/luohang/speech_lan/bilingual两个文件夹。

6 推荐读物

http://kaldi-asr.org/doc/
https://www.inf.ed.ac.uk/teaching/courses/asr/
http://deeplearning.net/reading-list/
Automatic Speech Recognition A Deep Learning Approcach By Dong Yu and Li Deng
Deep Learning by Yoshua Bengio,Ian Goodfellow,Aaron Courville

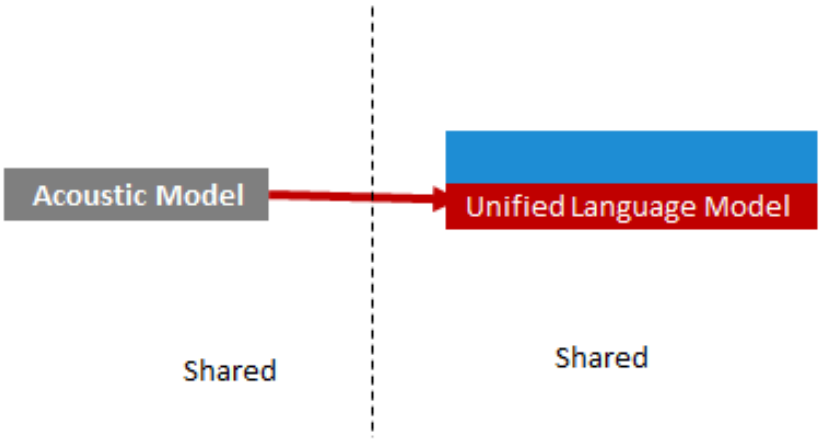

Figure 5: 统一的语言模型

Author details

[1]Center for Speech and Language Technologies, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China. [2]Center for Speech and Language Technologies, Division of Technical Innovation and Development, Tsinghua National Laboratory for Information Science and Technology, ROOM 1-303, BLDG FIT, 100084 Beijing, China. [3]Chengdu Institute of Computer Applications, Chinese Academy of Sciences, 610041 Chengdu, China.

References

1. Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdelrahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, and Tara N. Sainath. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.
2. Yun Lei, Nicolas Scheffer, Luciana Ferrer, and Mitchell Mclaren. A novel scheme for speaker recognition using a phonetically-aware deep neural network. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1695–1699, 2014.
3. Zhiyuan Tang, Lantian Li, and Dong Wang. Multi-task recurrent model for speech and speaker recognition. 2016.
4. Zhiyuan Tang, Lantian Li, and Dong Wang. Multi-task recurrent model for true multilingual speech recognition. 2016.
5. Hasim Sak, Andrew Senior, and Francoise Beaufays. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *Computer Science*, 2014.