ISOLATED WORD RECOGNITION IN KALDI

Author : Abhishek Dey

Aim : To build a two-class isolated word recognition system

Step 1: Create a working directory inside **kaldi-master/egs** directory. Say **Iso-ASR.**

```
cd  kaldi-master/egs

mkdir Iso-ASR
```

Step 2: Copy the following scripts & folders inside Iso-ASR directory.

- steps
- local
- utils
- conf
- run.sh
- fst.sh
- cmd.sh
- path.sh
- wav.sh

N.B :Set your **cluster options in cmd.sh** file and change the **KALDI_ROOT in path.sh** file

```
export KALDI_ROOT=/home/Kaldi_Workspace/kaldi-master
```

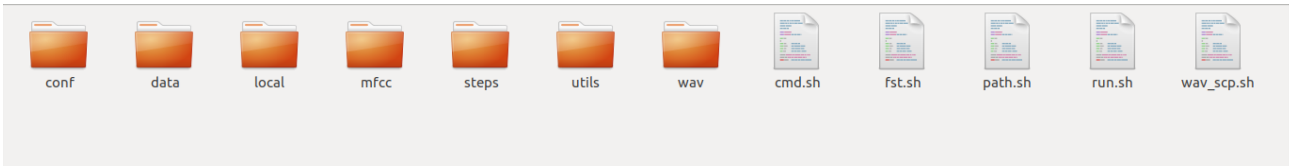
Step 3: Create a directory named **wav** inside Iso-ASR directory. Further create **two** directories named **train** and **test** inside wav directory. Copy all your training .wav files inside train directory and testing .wav files inside test directory.

```
cd Iso-ASR
mkdir wav
cd wav
mkdir train test
```

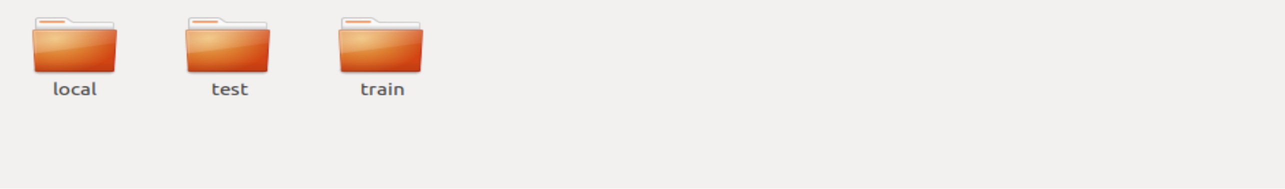
Step 4: Create a directory named **data** inside Iso-ASR directory. Further create **three** directories inside data directory named **train test** and **local.**

```
cd Iso-ASR
mkdir data
cd data
mkdir train test local
```

Your directory structure inside **Iso-ASR** should look like this at this step



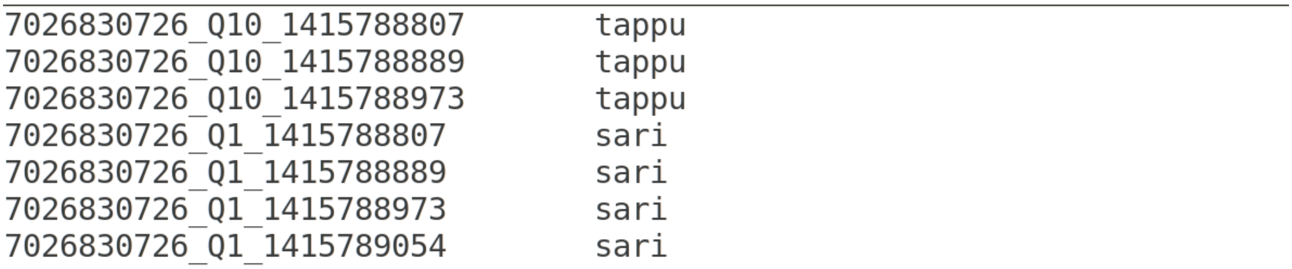
And inside **data** directory, it should look like this at this step



<u>Step 5</u>: You need to create 4 files inside **data/train** and **data/test** directories w.r.t training and testing sets.

- **text**–file containing the filename & their corresponding transcriptions

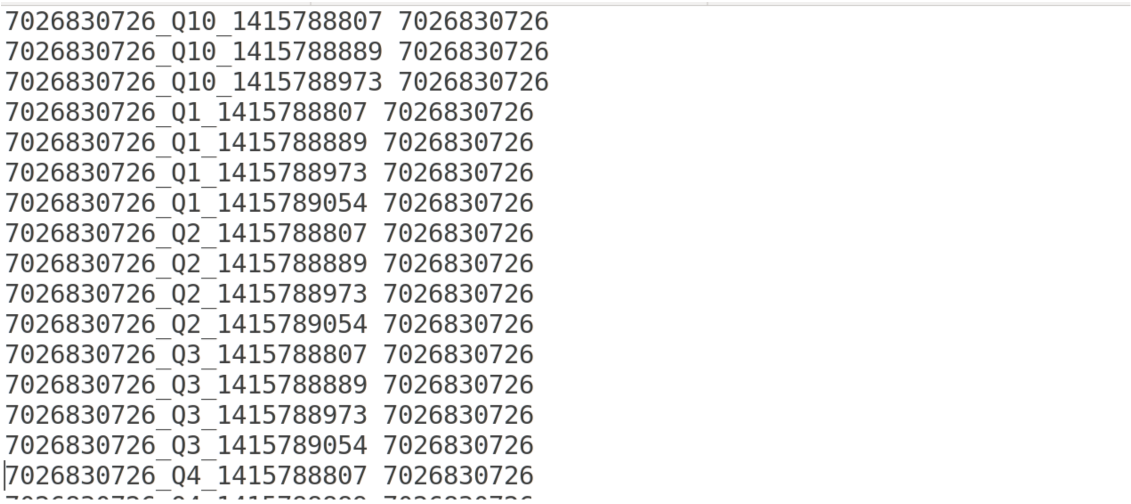
<Filename><tab><transcription>



```
7026830726_Q10_1415788807       tappu
7026830726_Q10_1415788889       tappu
7026830726_Q10_1415788973       tappu
7026830726_Q1_1415788807        sari
7026830726_Q1_1415788889        sari
7026830726_Q1_1415788973        sari
7026830726_Q1_1415789054        sari
```

- **utt2spk –** utterance to speaker mapping file

<utterance><tab><speaker>



```
7026830726_Q10_1415788807 7026830726
7026830726_Q10_1415788889 7026830726
7026830726_Q10_1415788973 7026830726
7026830726_Q1_1415788807 7026830726
7026830726_Q1_1415788889 7026830726
7026830726_Q1_1415788973 7026830726
7026830726_Q1_1415789054 7026830726
7026830726_Q2_1415788807 7026830726
7026830726_Q2_1415788889 7026830726
7026830726_Q2_1415788973 7026830726
7026830726_Q2_1415789054 7026830726
7026830726_Q3_1415788807 7026830726
7026830726_Q3_1415788889 7026830726
7026830726_Q3_1415788973 7026830726
7026830726_Q3_1415789054 7026830726
7026830726_Q4_1415788807 7026830726
```

cd data/train

Extract utterance list from the text file

cat text | awk ‘{print $1}’ > utt

Extract corresponding speaker list from the text file

cat text | cut -d ‘_’ -f1 > spk

Join utt and spk files side by side and create utt2spk file

paste utt spk > utt2spk

- **wav.scp** – file containing the filename & their corresponding wav file path

<Filename><tab><wav file path>

```
7026830726_Q10_1415788807       /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q10_14
7026830726_Q10_1415788889       /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q10_14
7026830726_Q10_1415788973       /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q10_14
7026830726_Q1_1415788807        /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q1_141
7026830726_Q1_1415788889        /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q1_141
7026830726_Q1_1415788973        /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q1_141
7026830726_Q1_1415789054        /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q1_141
7026830726_Q2_1415788807        /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q2_141
7026830726_Q2_1415788889        /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q2_141
7026830726_Q2_1415788973        /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q2_141
7026830726_Q2_1415789054        /home/Kaldi_Workspace/kaldi-master/egs//Iso-ASR/wav/train/7026830726_Q2_141
```

Open **wav_scp.sh** script

set **wav_path** as path containing the train/test wav files and **data_path** as path of data/train or data/test directories.

```
#!/bin/bash

# Author : Abhishek Dey

wav_path=/home/Kaldi_Workspace/kaldi-master/egs/Iso-ASR/wav/train

data_path=/home/Kaldi_Workspace/kaldi-master/egs/Iso-ASR/data/train

cat $data_path/utt | awk '{printf "%s\t%s%s%s\n",$1,"'$wav_path'/",$1,".wav"}' > $data_path/wav.scp
```

Then execute this script.

cd Iso-ASR

./wav_scp.sh

- **spk2utt** – speaker to utterance mapping fule

<Speaker><space><Utterance><space><Utterance>

```
7026830726 7026830726_Q10_1415788807 7026830726_Q10_1415788889 7026830726_Q10_1415788973
7026830726_Q1_1415788807 7026830726_Q1_1415788889 7026830726_Q1_1415788973 7026830726_Q1_1415789054
7026830726_Q2_1415788807 7026830726_Q2_1415788889 7026830726_Q2_1415788973 7026830726_Q2_1415789054
7026830726_Q3_1415788807 7026830726_Q3_1415788889 7026830726_Q3_1415788973 7026830726_Q3_1415789054
7026830726_Q4_1415788807 7026830726_Q4_1415788889 7026830726_Q4_1415788973 7026830726_Q4_1415789054
7026830726_Q5_1415788807 7026830726_Q5_1415788889 7026830726_Q5_1415788973 7026830726_Q5_1415789054
7026830726_Q6_1415788807 7026830726_Q6_1415788889 7026830726_Q6_1415788973 7026830726_Q7_1415788807
7026830726_Q7_1415788889 7026830726_Q7_1415788973 7026830726_Q8_1415788807 7026830726_Q8_1415788889
7026830726_Q8_1415788973 7026830726_Q9_1415788807 7026830726_Q9_1415788889 7026830726_Q9_1415788973
7204582600 7204582600_Q10_1415677945 7204582600_Q10_1415678028 7204582600_Q10_1415678111
7204582600_Q10_1415678194 7204582600_Q10_1415678530 7204582600_Q1_1415677945 7204582600_Q1_1415678028
7204582600_Q1_1415678111 7204582600_Q1_1415678194 7204582600_Q1_1415678277 7204582600_Q1_1415678530
7204582600_Q2_1415677945 7204582600_Q2_1415678028 7204582600_Q2_1415678111 7204582600_Q2_1415678194
7204582600_Q2_1415678277 7204582600_Q2_1415678530 7204582600_Q3_1415677945 7204582600_Q3_1415678028
7204582600_Q3_1415678111 7204582600_Q3_1415678194 7204582600_Q3_1415678277 7204582600_Q3_1415678530
7204582600_Q4_1415677945 7204582600_Q4_1415678028 7204582600_Q4_1415678111 7204582600_Q4_1415678194
7204582600_Q4_1415678277 7204582600_Q4_1415678530 7204582600_Q5_1415677945 7204582600_Q5_1415678028
```

cd data/train

../../utils/utt2spk_to_spk2utt.pl utt2spk > spk2utt

<u>Step 6 :</u> Create a dictionary directory **(say dict_yesno)** insde data/local directory

- **extra_questions.txt** (This file is kept blank)
- **extra_phones.txt** (This file is kept blank)
- **lexicon.txt** (Word & its phone level break up)

Since in this case we will be building **word models, we will have word to word mapping.**

sari	sari
tappu	tappu
sil	sil
!SIL	sil

- **nonsilence_phones.txt** (All the phones excluding silence)

  sari
  tappu

- **phones.txt** (All the phones including silence)

  sari
  tappu
  sil

- **optional_silence.txt**(silence phone)
- **silence_phones.txt** (silence phone including additional fillers such as bgnoise,chnoise)

In our case **optional_silence.txt** & **silence_phones.txt** are same since we haven't used additional fillers such as backgroud noise, channel noise etc.

<u>Step</u> 7: Now we will create language models

Note that you shoud be a root user.

Sudo su mode

set paths in **fst.sh** script

for x in **yesno**

do

lang_dir=/home/Kaldi_Workspace/kaldi-master/egs/Iso-ASR/data/lang_$x
tmp_dir=/home/Kaldi_Workspace/kaldi-master/egs/Iso-ASR/data/local/tmp_$x

done

Now Run

<DOT><SPACE>path.sh
./fst.sh

This should give a **SUCCESS** Message after running.

```
Checking word_boundary.int and disambig.int
--> generating a 91 words sequence
--> resulting phone sequence from L.fst corresponds to the word sequence
--> L.fst is OK
--> generating a 98 words sequence
--> resulting phone sequence from L_disambig.fst corresponds to the word sequence
--> L_disambig.fst is OK

Checking /home/asr/Desktop/kaldi-trunk/egs/Tamil_demo/data/lang_test/oov.{txt, int} ...
--> 1 entry/entries in /home/asr/Desktop/kaldi-trunk/egs/Tamil_demo/data/lang_test/oov.txt
--> /home/asr/Desktop/kaldi-trunk/egs/Tamil_demo/data/lang_test/oov.int corresponds to /home/asr/Desktop/kaldi-trunk/eg
s/Tamil_demo/data/lang_test/oov.txt
--> /home/asr/Desktop/kaldi-trunk/egs/Tamil_demo/data/lang_test/oov.{txt, int} are OK

--> SUCCESS
root@ASR:/home/asr/Desktop/kaldi-trunk/egs/workspace#
```

This Creates **G.fst** in **data/lang_test** and **data/lang_train** directory. To Check the memory of **G.fst** do the following:

du -hsc data/lang_test/G.fst

Step 8: Now open the script. **run.sh .** You need to set the switches.

```
train_nj=10
decode_nj=8

#================================
#        SET SWITCHES
#================================

mfcc_extract_sw=0

mono_train_sw=0
mono_test_sw=1

tri1_train_sw=0
tri1_test_sw=1

tri2_train_sw=0
tri2_test_sw=1

tri3_train_sw=0
tri3_test_sw=1

sgmm_train_sw=0
sgmm_test_sw=1

dnn_train_sw=0
dnn_test_sw=1
```

train_nj & **decode_nj** indicate the **number of jobs** during **training** & **decoding** respectively. Here train_nj 10 means the whole job will be divided into 10 parts. Based on the Processor you can change these parameters.

You need to set the directories

train_dir1=data/train
train_lang_dir=data/lang_yesno

test_dir1=data/test
test_lang_dir1=data/lang_yesno

graph_dir1=graph_yesno
decode_dir1=decode_yesno

At first the script computes mfcc

```
root@ASR: /home/asr/Desktop/kaldi-trunk/egs/workspace
============================================================================
         MFCC Feature Extration & CMVN for Training and Test set
============================================================================
steps/make_mfcc.sh --cmd run.pl --nj 3 data/train exp/make_mfcc/train mfcc
Successfully validated data-directory data/train
steps/make_mfcc.sh: [info]: no segments file exists: assuming wav.scp indexed by ut
terance.
```

- Then it performs **Monophone Training & Decoding**

```
Succeeded creating CMVN stats for train
steps/make_mfcc.sh --cmd run.pl --nj 3 data/test exp/make_mfcc/test mfcc
Successfully validated data-directory data/test
steps/make_mfcc.sh: [info]: no segments file exists: assuming wav.scp indexed by ut
terance.
Succeeded creating MFCC features for test
steps/compute_cmvn_stats.sh data/test exp/make_mfcc/test mfcc
Succeeded creating CMVN stats for test
===============================================================================
                      MonoPhone Training & Decoding
===============================================================================
steps/train_mono.sh --nj 3 --cmd run.pl data/train data/lang exp/mono
steps/train_mono.sh: Initializing monophone system.
steps/train_mono.sh: Compiling training graphs
steps/train_mono.sh: Aligning data equally (pass 0)
steps/train_mono.sh: Pass 1
steps/train_mono.sh: Aligning data
steps/train_mono.sh: Pass 2
steps/train_mono.sh: Aligning data
```

- Two new folders are created in **workspace** : **exp** & **mfcc. exp contain the training models & decoded outputs.**

- **The Word Error Rate(WER) files are located in**

  exp/mono/decode_yesno

- **The decoded outputs are located in**

  exp/mono/decode/_yesno/log

- **Command to check all the WER**

  cat/exp/mono/decode_yesno/wer_*| egrep WER | sort -n

****