# Kaldi

Weida Wang, 2151300

## Install Kaldi (in Linux)

1. Download the GitHub repository

   ```
   git clone https://github.com/kaldi-asr/kaldi
   ```

2. Compile dependent toolkits in kaldi/tools

   Install subversion, automake, autoconf, libtool, g++, zlib, libatal, wget, sox, gfortran

   ```
   sudo apt-get install subversion automake autoconf libtool g++ zlib1g-dev
   libatlas-base-dev wget sox gfortran
   ```

   Install MKL for faster linear algebra computations

   ```
   cd tools
   bash extras/install_mkl.sh
   ```

   Continue until the following command runs, and the appearance of the image below indicates that all dependencies are installed:

   ```
   extras/check_dependencies.sh
   ```

   ```
   root@autodl-container-81914db329-c1c49fae:~/kaldi/tools# extras/check_dependencies.sh
   extras/check_dependencies.sh: all OK.
   ```

   Speed up the compilation of `kaldi/tools` using multiple processes:

   ```
   make -j 4
   ```

   Install openfst

   ```
   make openfst
   ```

3. Compile Kaldi's core libraries `kaldi/src`

   Run the configuration script

   ```
   cd ../src
   ./configure --shared
   ```

   The final compilation

   ```
   make depend -j 4
   make -j 4
   ```

# Run yesno and THCHS-30 examples

## Test yes/no example

Requires internet connection as it involves downloading a small amount of data

```
cd ../egs/yesno/s5
sh run.sh
```

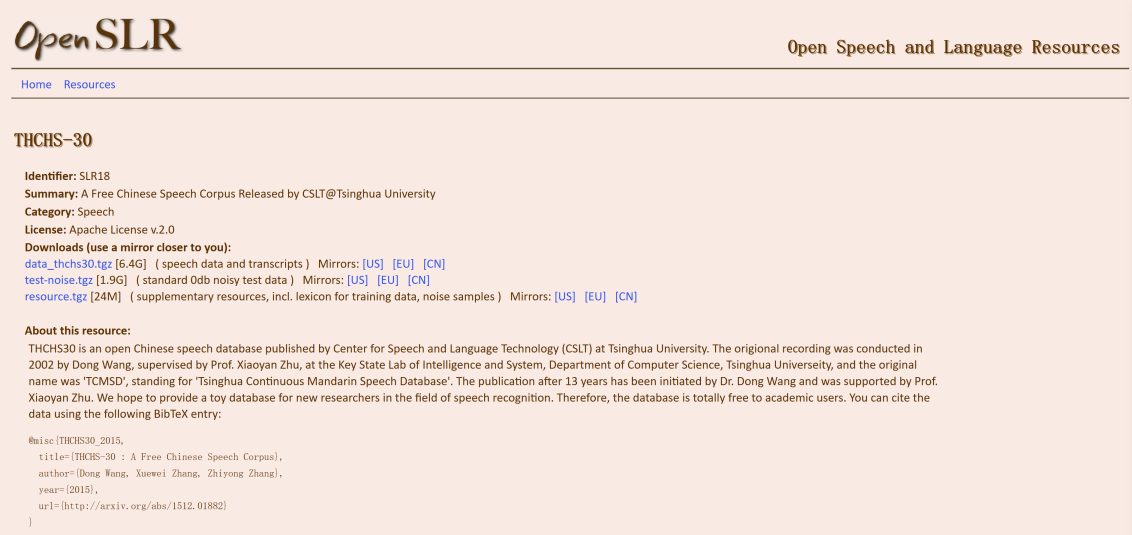
```
root@autodl-container-81914db329-c1c49fae:~/kaldi/egs/yesno/s5# sh run.sh
--2023-11-30 10:00:21--  http://www.openslr.org/resources/1/waves_yesno.tar.gz
Resolving www.openslr.org (www.openslr.org)... 46.101.158.64
Connecting to www.openslr.org (www.openslr.org)|46.101.158.64|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://openslr.magicdatatech.com/resources/1/waves_yesno.tar.gz [following]
--2023-11-30 10:00:23--  https://openslr.magicdatatech.com/resources/1/waves_yesno.tar.gz
Resolving openslr.magicdatatech.com (openslr.magicdatatech.com)... 39.96.249.211
Connecting to openslr.magicdatatech.com (openslr.magicdatatech.com)|39.96.249.211|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4703754 (4.5M) [application/x-gzip]
Saving to: ‘waves_yesno.tar.gz’

waves_yesno.tar.gz                  87%[=============================================================================>            ]   3.91M  --.-KB/s    eta 11s
```

If the run is successful, the WER (Word Error Rate) metric will appear as shown in the following image:

```
local/score.sh --cmd utils/run.pl data/test_yesno exp/mono0a/graph_tgpr exp/mono0a/decode_test_yesno
local/score.sh: scoring with word insertion penalty=0.0,0.5,1.0
%WER 0.00 [ 0 / 232, 0 ins, 0 del, 0 sub ] exp/mono0a/decode_test_yesno/wer_10_0.0
```

## Test THCHS-30 example

1. Download thchs30 from https://www.openslr.org/18/.




2. Since I am using a cloud server on AutoDL, it is necessary to upload the data from the local machine to the cloud server. I recommend using AutoPannal for uploading, it's super fast! As shown in the following image：Link


   Then use the following command to unzip the three tar packages into one folder

```
sudo tar -zxvf /root/autodl-tmp/resource.tgz
```

3. Since we are training on a single machine and not on an Oracle GridEngine cluster, change all queues in `egs/thchs30/s5/cmd.sh` to run

```
export train_cmd=run.pl
export decode_cmd="run.pl --mem 4G"
export mkgraph_cmd="run.pl --mem 8G"
export cuda_cmd="run.pl --gpu 1"
```

4. Modify the number of parallel jobs in `run.sh`

```
n=8      #parallel jobs 原则上要小于0.7*物理cpu数*单cpu核数
```

Modify the path in `run.sh` to match the location of our dataset

```
# thchs=/nfs/public/materials/data/thchs30-openslr
# my dataset root: /root/autodl-tmp/data_thchs30
thchs=/root/autodl-tmp/thchs30
```

According the referenced PPT，we just need to test these four models and `comment` the rest tests in run.sh.

```
#monophone
steps/train_mono.sh --boost-silence 1.25 --nj $n --cmd "$train_cmd" data/mfcc/train data/lang exp/mono || exit 1;
#test monophone model
local/thchs-30_decode.sh --mono true --nj $n "steps/decode.sh" exp/mono data/mfcc &

#monophone_ali
steps/align_si.sh --boost-silence 1.25 --nj $n --cmd "$train_cmd" data/mfcc/train data/lang exp/mono exp/mono_ali || exit 1;

#triphone
steps/train_deltas.sh --boost-silence 1.25 --cmd "$train_cmd" 2000 10000 data/mfcc/train data/lang exp/mono_ali exp/tri1 || exit 1;
#test tri1 model
local/thchs-30_decode.sh --nj $n "steps/decode.sh" exp/tri1 data/mfcc &

#triphone_ali
steps/align_si.sh --nj $n --cmd "$train_cmd" data/mfcc/train data/lang exp/tri1 exp/tri1_ali || exit 1;

#lda_mllt
steps/train_lda_mllt.sh --cmd "$train_cmd" --splice-opts "--left-context=3 --right-context=3" 2500 15000 data/mfcc/train data/lang exp/tri1_ali exp/tri2b || exit 1;
#test tri2b model
local/thchs-30_decode.sh --nj $n "steps/decode.sh" exp/tri2b data/mfcc &


#lda_mllt_ali
steps/align_si.sh  --nj $n --cmd "$train_cmd" --use-graphs true data/mfcc/train data/lang exp/tri2b exp/tri2b_ali || exit 1;

#sat
steps/train_sat.sh --cmd "$train_cmd" 2500 15000 data/mfcc/train data/lang exp/tri2b_ali exp/tri3b || exit 1;
#test tri3b model
local/thchs-30_decode.sh --nj $n "steps/decode_fmllr.sh" exp/tri3b data/mfcc &
```

5. Check the experimental results

It is recommended to redirect the output from the terminal to a log file with the following code at the beginning of `run.sh`:

```
exec > >(tee log.log) 2>&1
```

- **Monophone Training**

  The end of the monophone training output summarizes the completion of the monophone model training and provides statistics such as the average alignment probability, total training time, percentages of retries and failures, as well as the number of states and Gaussian mixtures.

  ```
  exp/mono: nj=8 align prob=-100.09 over 25.49h [retry=0.2%, fail=0.0%]
  states=656 gauss=989
  ```

```
steps/train_mono.sh: Pass 35
steps/train_mono.sh: Aligning data
steps/train_mono.sh: Pass 36
steps/train_mono.sh: Pass 37
steps/train_mono.sh: Pass 38
steps/train_mono.sh: Aligning data
steps/train_mono.sh: Pass 39
steps/diagnostic/analyze_alignments.sh --cmd run.pl data/lang exp/mono
steps/diagnostic/analyze_alignments.sh: see stats in exp/mono/log/analyze_alignments.log
3978 warnings in exp/mono/log/align.*.*.log
55 warnings in exp/mono/log/acc.*.*.log
1052 warnings in exp/mono/log/update.*.log
exp/mono: nj=8 align prob=-100.09 over 25.49h [retry=0.2%, fail=0.0%] states=656 gauss=989
steps/train_mono.sh: Done training monophone system in exp/mono
```

- **Triphone Training**

```
exp/tri1: nj=8 align prob=-96.75 over 25.49h [retry=0.3%, fail=0.0%]
states=1664 gauss=10023 tree-impr=4.80
```

```
steps/train_deltas.sh: training pass 31
steps/train_deltas.sh: training pass 32
steps/train_deltas.sh: training pass 33
steps/train_deltas.sh: training pass 34
steps/diagnostic/analyze_alignments.sh --cmd run.pl data/lang exp/tri1
steps/diagnostic/analyze_alignments.sh: see stats in exp/tri1/log/analyze_alignments.log
1 warnings in exp/tri1/log/compile_questions.log
81 warnings in exp/tri1/log/align.*.*.log
1 warnings in exp/tri1/log/build_tree.log
281 warnings in exp/tri1/log/update.*.log
9 warnings in exp/tri1/log/questions.log
59 warnings in exp/tri1/log/acc.*.*.log
9 warnings in exp/tri1/log/init_model.log
exp/tri1: nj=8 align prob=-96.75 over 25.49h [retry=0.3%, fail=0.0%] states=1664 gauss=10023 tree-impr=4.80
steps/train_deltas.sh: Done training system with delta+delta-delta features in exp/tri1
```

- **LDA+MLLT Training**

```
exp/tri2b: nj=8 align prob=-48.18 over 25.48h [retry=0.6%, fail=0.0%]
states=2064 gauss=15038 tree-impr=4.32 lda-sum=23.97
mllt:impr,logdet=1.21,1.71
```

```
Training pass 33
Training pass 34
steps/diagnostic/analyze_alignments.sh --cmd run.pl data/lang exp/tri2b
steps/diagnostic/analyze_alignments.sh: see stats in exp/tri2b/log/analyze_alignments.log
2 warnings in exp/tri2b/log/lda_acc.*.log
147 warnings in exp/tri2b/log/align.*.*.log
1 warnings in exp/tri2b/log/compile_questions.log
8 warnings in exp/tri2b/log/questions.log
128 warnings in exp/tri2b/log/acc.*.*.log
1 warnings in exp/tri2b/log/build_tree.log
272 warnings in exp/tri2b/log/update.*.log
9 warnings in exp/tri2b/log/init_model.log
exp/tri2b: nj=8 align prob=-48.18 over 25.48h [retry=0.6%, fail=0.0%] states=2064 gauss=15038 tree-impr=4.32 lda-sum=23.97 mllt:impr,logdet=1.21,1.71
steps/train_lda_mllt.sh: Done training system with LDA+MLLT features in exp/tri2b
```

- **SAT Training**

```
exp/tri3b: nj=8 align prob=-47.92 over 25.48h [retry=0.5%, fail=0.0%]
states=2096 gauss=15025 fmllr-impr=2.43 over 18.95h tree-impr=6.43
```

```
Pass 33
Pass 34
steps/diagnostic/analyze_alignments.sh --cmd run.pl data/lang exp/tri3b
steps/diagnostic/analyze_alignments.sh: see stats in exp/tri3b/log/analyze_alignments.log
180 warnings in exp/tri3b/log/align.*.*.log
272 warnings in exp/tri3b/log/update.*.log
1 warnings in exp/tri3b/log/compile_questions.log
8 warnings in exp/tri3b/log/questions.log
1 warnings in exp/tri3b/log/build_tree.log
8 warnings in exp/tri3b/log/est_alimdl.log
156 warnings in exp/tri3b/log/acc.*.*.log
21 warnings in exp/tri3b/log/fmllr.*.*.log
9 warnings in exp/tri3b/log/init_model.log
steps/train_sat.sh: Likelihood evolution:
-49.7349 -49.5281 -49.4331 -49.3207 -48.6793 -48.1404 -47.8006 -47.567 -47.3919 -46.9426 -46.7473 -46.5429 -46.4248 -46.3218 -46.2255 -46
exp/tri3b: nj=8 align prob=-47.92 over 25.48h [retry=0.5%, fail=0.0%] states=2096 gauss=15025 fmllr-impr=2.43 over 18.95h tree-impr=6.43
steps/train_sat.sh: done training SAT system in exp/tri3b
```

# Recognize my own speech

1. Install PortAudio

```
cd tools
./install_portaudio.sh
```

安装成功显示如下：

```
---------------------------------------------------------------
PortAudio was successfully installed.

On some systems (e.g. Linux) you should run 'ldconfig' now
to make the shared object available.  You may also need to
modify your LD_LIBRARY_PATH environment variable to include
the directory /root/kaldi/tools/portaudio/install/lib
---------------------------------------------------------------
```

2. Compile Related Tools:

When the compilation is complete, it should display `Done`

```
cd ..
cd src
make ext
```

3. Record Your Own Audio:

You can use the provided Python code to record your audio and generate your `mine.wav` file. Make sure you have the required libraries installed.

```
import pyaudio
import wave

def record_audio(output_filename="output.wav", record_seconds=3):
    # Basic parameter settings
    CHUNK = 1024  # Number of frames per buffer
    FORMAT = pyaudio.paInt16  # Size and format of each sample
    CHANNELS = 1  # Number of audio channels (1 for mono, 2 for stereo)
    RATE = 16000   # Sampling rate in Hz

    p = pyaudio.PyAudio()

    # Start recording
    print("Recording started...")
    stream = p.open(format=FORMAT,
                    channels=CHANNELS,
                    rate=RATE,
                    input=True,
                    frames_per_buffer=CHUNK)

    frames = []

    for _ in range(0, int(RATE / CHUNK * record_seconds)):
        data = stream.read(CHUNK)
        frames.append(data)
```

```
    print("Recording finished")

    # Stop recording
    stream.stop_stream()
    stream.close()
    p.terminate()

    # Save as a WAV file
    wf = wave.open(output_filename, 'wb')
    wf.setnchannels(CHANNELS)
    wf.setsampwidth(p.get_sample_size(FORMAT))
    wf.setframerate(RATE)
    wf.writeframes(b''.join(frames))
    wf.close()

    print(f"Audio saved as {output_filename}")

if __name__ == "__main__":
    record_audio("mine.wav", 10)
```

4. Modify the Script `oneline_demo/run.sh`
   - Comment out the file downloading section

```
#if [ ! -s ${data_file}.tar.bz2 ]; then
#    echo "Downloading test models and data ..."
#    wget -T 10 -t 3 $data_url;

#    if [ ! -s ${data_file}.tar.bz2 ]; then
#        echo "Download of $data_file has failed!"
#        exit 1
#    fi
#fi
```

   - Change the acoustic model type `ac_model_type` to `tri1`。

```
ac_model_type=tri1
```

   - Modify the recognition code to use your recorded audio:

```
simulated)
    echo
    echo -e "  SIMULATED ONLINE DECODING - pre-recorded audio is used\n"
    echo "  The (bigram) language model used to build the decoding graph
was"
    echo "  estimated on an audio book's text. The text in question is"
    echo "  \"King Solomon's Mines\"
(http://www.gutenberg.org/ebooks/2166)."
    echo "  The audio chunks to be decoded were taken from the audio
book read"
    echo "  by John Nicholson(http://librivox.org/king-solomons-mines-
by-haggard/)"
    echo
```

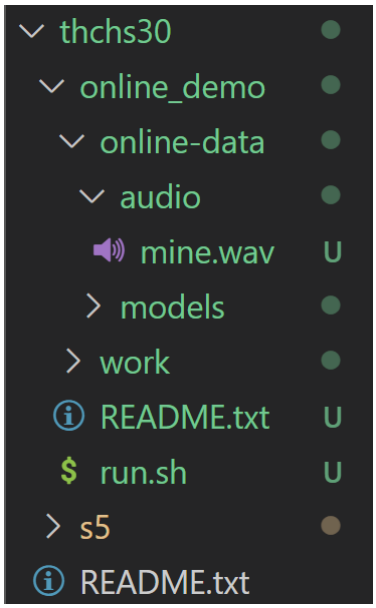
```
    echo "  NOTE: Using utterances from the book, on which the LM was
estimated"
    echo "        is considered to be \"cheating\" and we are doing this
only for"
    echo "        the purposes of the demo."
    echo
    echo "  You can type \"./run.sh --test-mode live\" to try it using
your"
    echo "  own voice!"
    echo
    mkdir -p $decode_dir
    # make an input .scp file
    > $decode_dir/input.scp
    for f in $audio/*.wav; do
        bf=`basename $f`
        bf=${bf%.wav}
        echo $bf $f >> $decode_dir/input.scp
    done
#    online-wav-gmm-decode-faster --verbose=1 --rt-min=0.8 --rt-
max=0.85\
#        --max-active=4000 --beam=12.0 --acoustic-scale=0.0769 \
#        scp:$decode_dir/input.scp $ac_model/model $ac_model/HCLG.fst \
#        $ac_model/words.txt '1:2:3:4:5' ark,t:$decode_dir/trans.txt \
#        ark,t:$decode_dir/ali.txt $trans_matrix;;
    online-wav-gmm-decode-faster  --verbose=1 --rt-min=0.8 --rt-max=0.85
--max-active=4000 \
       --beam=12.0 --acoustic-scale=0.0769 --left-context=3 --right-
context=3 \
       scp:$decode_dir/input.scp $ac_model/final.mdl $ac_model/HCLG.fst
\
       $ac_model/words.txt '1:2:3:4:5' ark,t:$decode_dir/trans.txt \
       ark,t:$decode_dir/ali.txt $trans_matrix;;
```

5. Create Directory Structure and Sync Recognition Model:
    1. Copy the "online_demo" directory from "voxforge" to the same level as "thchs30/s5".
    2. Inside the "online_demo" directory, create "online-data" and "work" directories.
    3. Inside "online-data," create "audio" and "models" directories.
        - Place the audio file you want to recognize (i.e., "mine.wav") inside the "audio" directory.
        - Inside the "models" directory, create a "tri1" directory.

          Copy the following files from "s5/exp/tri1" to your "tri1" directory: `final.mdl` and `35.mdl`

          Copy the following files from "s5/exp/tri1/graph_word" to your "tri1" directory: `words.txt` and `HCLG.fst`

    Your directory structure should resemble the one you provided.




6. Perform Testing:

   In the terminal, run the following command to perform testing:

   Please note that the recognition accuracy may not be very high, and the recognized text may contain errors.

   My input is "*语音识别第三次作业。静夜思，床前明月光，疑是地上霜，举头望明月，低头思故乡。*", while output is:

```
  SIMULATED ONLINE DECODING - pre-recorded audio is used

  The (bigram) language model used to build the decoding graph was
  estimated on an audio book's text. The text in question is
  "King Solomon's Mines" (http://www.gutenberg.org/ebooks/2166).
  The audio chunks to be decoded were taken from the audio book read
  by John Nicholson(http://librivox.org/king-solomons-mines-by-haggard/)

  NOTE: Using utterances from the book, on which the LM was estimated
        is considered to be "cheating" and we are doing this only for
        the purposes of the demo.

  You can type "./run.sh --test-mode live" to try it using your
  own voice!

online-wav-gmm-decode-faster --verbose=1 --rt-min=0.8 --rt-max=0.85 --max-active=4000 --beam=12.0 --acoustic-scale=0.0769 --left-context=3 --right-context=3 scp
:./work/input.scp online-data/models/tri1/final.mdl online-data/models/tri1/HCLG.fst online-data/models/tri1/words.txt 1:2:3:4:5 ark,t:./work/trans.txt ark,t:./
work/ali.txt
File: mine
年 人事 费用 结算 昨日

拥有 四 耗损 如果 一时 的 手 说 之后 我 名人 系统 似 无效

./run.sh: line 106: online-data/audio/trans.txt: No such file or directory
./run.sh: line 111: gawk: command not found
compute-wer --mode=present ark,t:./work/ref.txt ark,t:./work/hyp.txt
WARNING (compute-wer[5.5.1124~1-21ae4]:Open():util/kaldi-table-inl.h:513) Failed to open stream ./work/ref.txt
ERROR (compute-wer[5.5.1124~1-21ae4]:SequentialTableReader():util/kaldi-table-inl.h:860) Error constructing TableReader: rspecifier is ark,t:./work/ref.txt

[ Stack-Trace: ]
/root/kaldi/src/lib/libkaldi-base.so(kaldi::MessageLogger::LogMessage() const+0x793) [0x7fe38b65a1c3]
compute-wer(kaldi::MessageLogger::LogAndThrow::operator=(kaldi::MessageLogger const&)+0x25) [0x55e4b0e55d91]
compute-wer(kaldi::SequentialTableReader<kaldi::TokenVectorHolder>::SequentialTableReader(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocato
r<char> > const&)+0xc4) [0x55e4b0e5c8da]
compute-wer(main+0x21e) [0x55e4b0e54d07]
/usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_main+0xf3) [0x7fe38b23f083]
compute-wer(_start+0x2e) [0x55e4b0e54a2e]
```