Building a Speech Recognition System with Kaldi

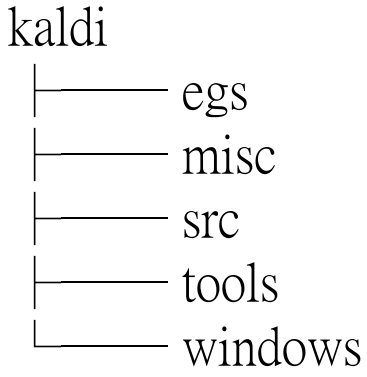
Compiled from:

- Sanjeev Khudanpur, Dan Povey and Jan Trmal, "Building Speech Recogni0on Systems with the Kaldi Toolkit", **https://www.clsp.jhu.edu/wp-content/uploads/sites/75/2016/06/Building-Speech-Recognition-Systems-with-the-Kaldi-Toolkit.pdf**
- Eleanor Chodroff, "Corpus Phonetics Tutorial/Kaldi", https://www.eleanorchodroff.com/tutorial/kaldi/kaldi-familiarization.html
- 篠崎隆宏, Kaldiツールキットを用いた音声認識システムの構築 - 東京工業大学, http://www.ts.ip.titech.ac.jp/demos/csjkaldisp2016oct.pdf

and many other sources.

Yuan-Fu Liao
National Taipei University of Technology
yfliao@ntutedu.tw

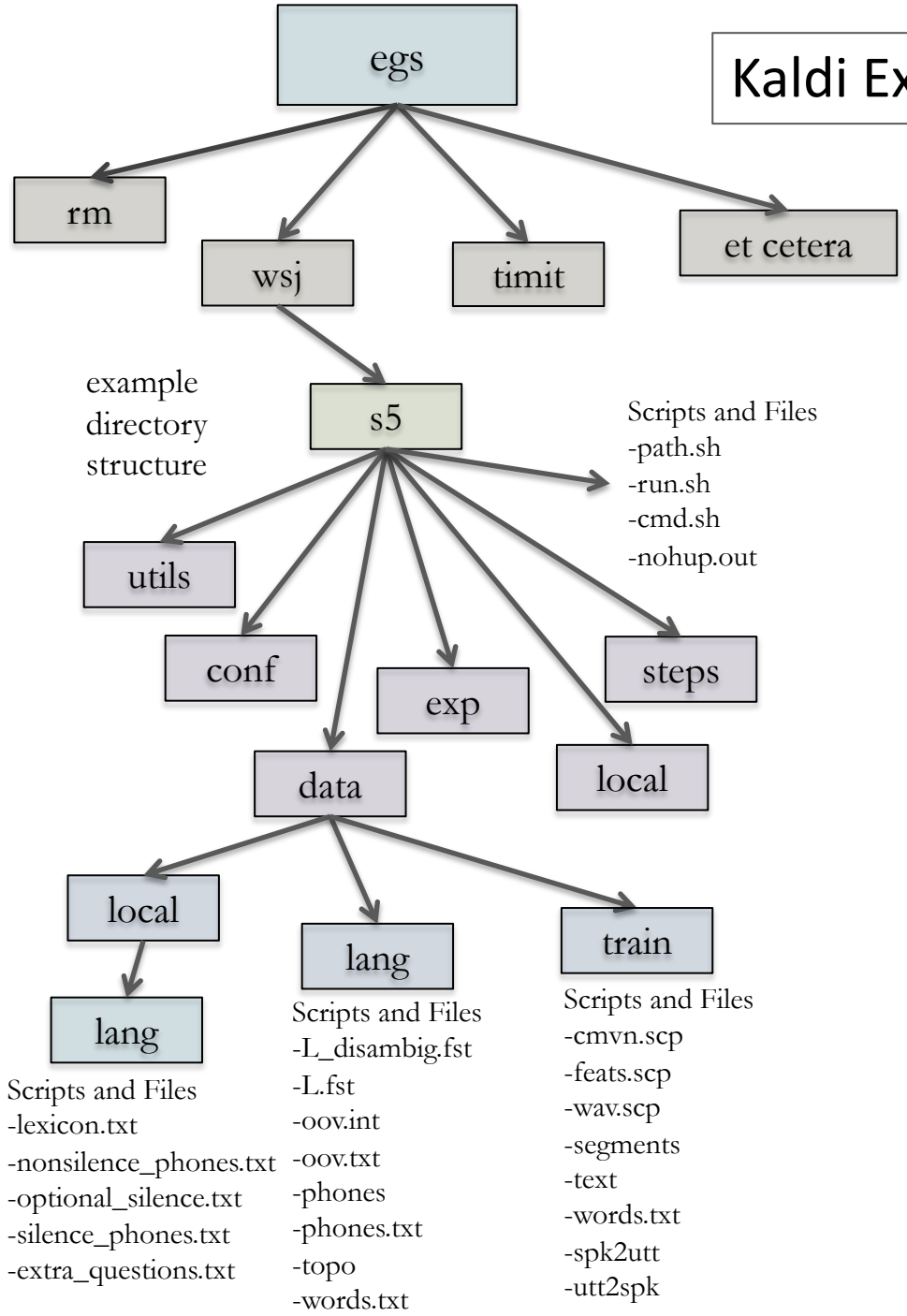
Kaldi Directory Structure



Building an STT System with Kaldi

- Directories Structure
  - kaldi/
    - tools/ (Installation scripts to install external tools)
    - src/ (The Kaldi source code)
      - base/, matrix/, util/, feat/, tree/, optimization/, gmm/, transform/, sgmm/, fstext/, hmm/, lm/, decoder/, bin/, fstbin/, gmmbin/, fgmmbin/, sgmmbin/, featbin/
    - egs/ (example scripts)
      - rm/s1/ (Resource Management example dir)
      - wsj/s1/ (Wall Street Journal example dir)
    - misc/ (additional tools and supplies)
    - Windows/ (tools for running Kaldi using Windows)

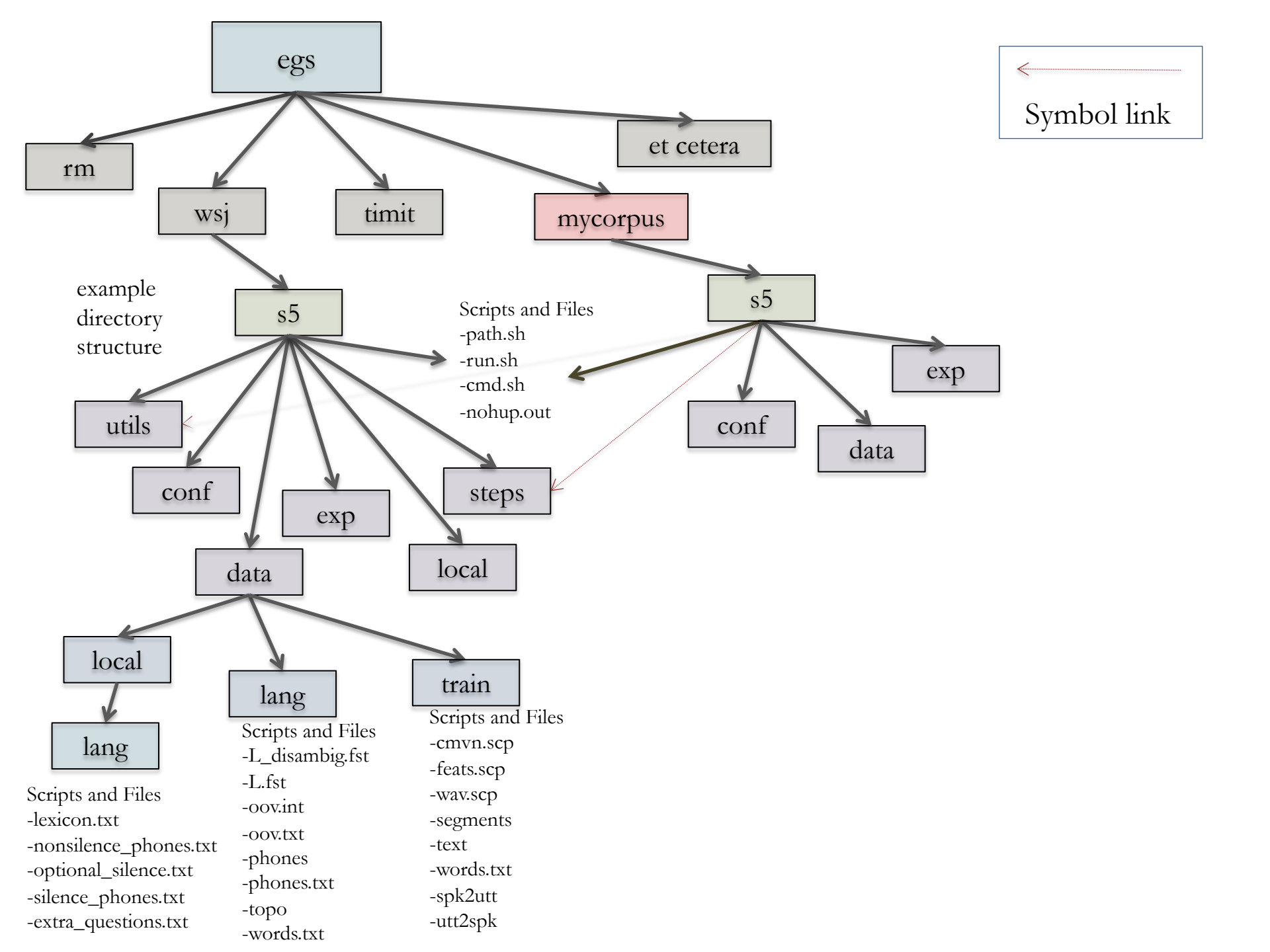
Kaldi Example Recipes Directory Structure

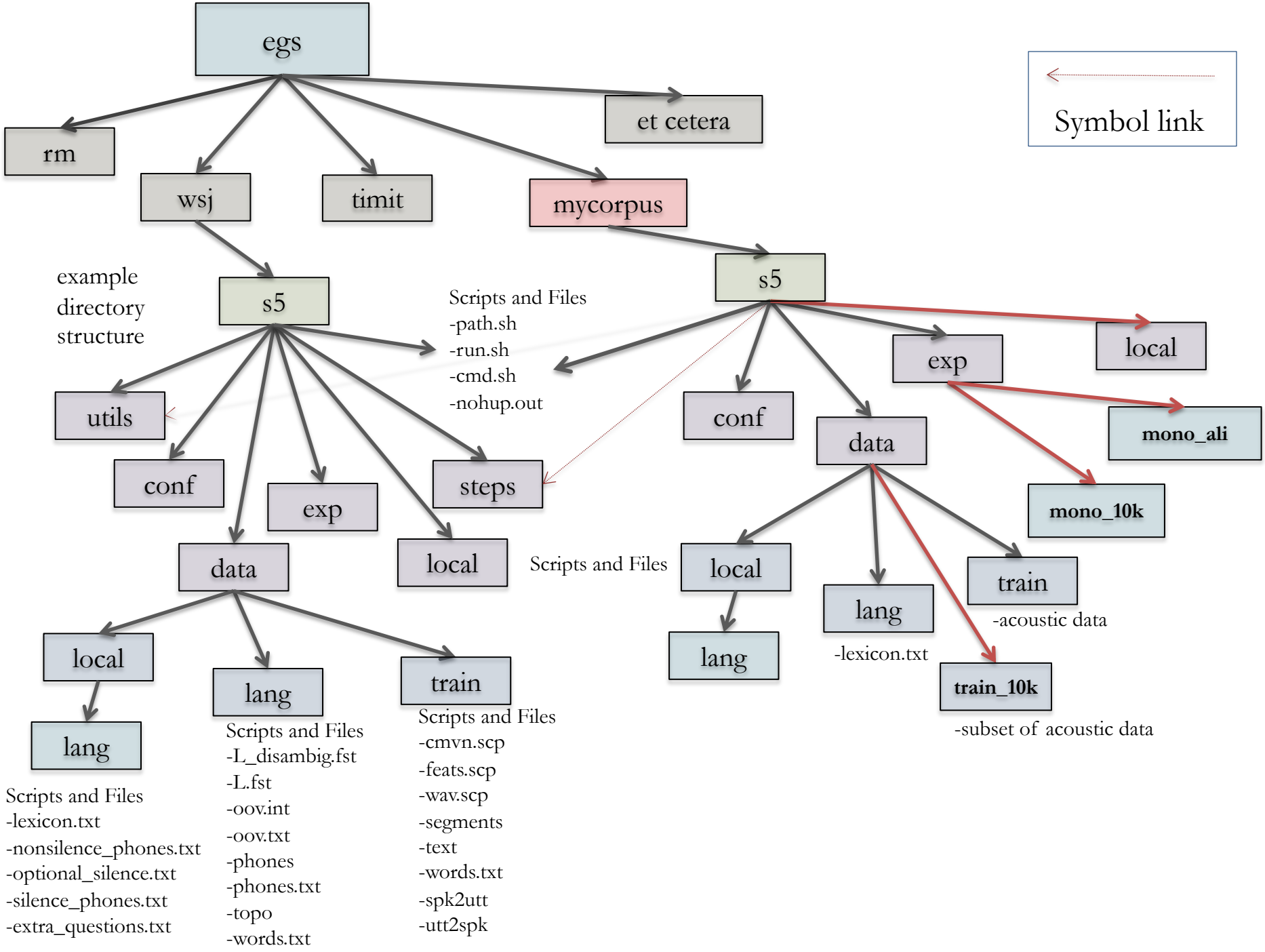


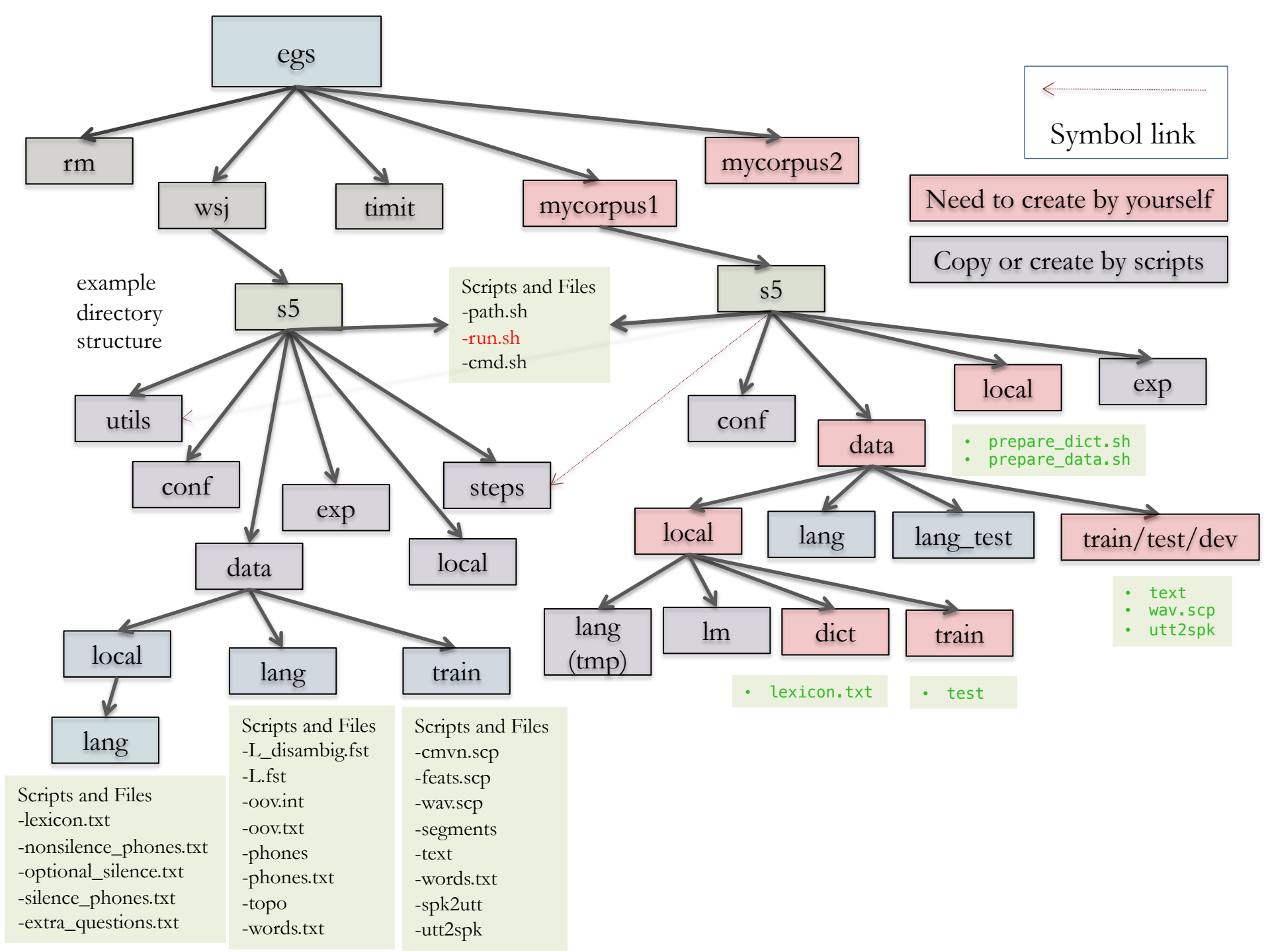
Build Your Own STT

- Copy/symbol link example scripts from egs/wsj/ to mycorpus/
- mycorpus/s5/

Files	Content
cmd.sh	Environment
path.sh	Path
run.sh	Scripts
data/	Corpus
@steps/, @utils/	Universal scripts (symbol link)
local/	Local scripts


egs
rm
wsj
timit
mycorpus
et cetera
Symbol link
example
directory
structure
s5
utils
conf
exp
steps
data
local
Scripts and Files
-path.sh
-run.sh
-cmd.sh
-nohup.out
s5
conf
data
exp
local
lang
lang
train
Scripts and Files
-lexicon.txt
-nonsilence_phones.txt
-optional_silence.txt
-silence_phones.txt
-extra_questions.txt
Scripts and Files
-L_disambig.fst
-L.fst
-oov.int
-oov.txt
-phones
-phones.txt
-topo
-words.txt
Scripts and Files
-cmvn.scp
-feats.scp
-wav.scp
-segments
-text
-words.txt
-spk2utt
-utt2spk


egs
rm
wsj
timit
mycorpus
et cetera
Symbol link
example
directory
structure
s5
utils
conf
exp
steps
data
local
lang
Scripts and Files
-path.sh
-run.sh
-cmd.sh
-nohup.out
Scripts and Files
-lexicon.txt
-nonsilence_phones.txt
-optional_silence.txt
-silence_phones.txt
-extra_questions.txt
Scripts and Files
-L_disambig.fst
-L.fst
-oov.int
-oov.txt
-phones
-phones.txt
-topo
-words.txt
train
Scripts and Files
-cmvn.scp
-feats.scp
-wav.scp
-segments
-text
-words.txt
-spk2utt
-utt2spk
s5
conf
data
exp
local
mono_ali
mono_10k
Scripts and Files
local
lang
lang
-lexicon.txt
train
-acoustic data
train_10k
-subset of acoustic data


egs
rm
wsj
timit
mycorpus1
mycorpus2
Symbol link
Need to create by yourself
Copy or create by scripts
example
directory
structure
s5
Scripts and Files
-path.sh
-run.sh
-cmd.sh
s5
utils
conf
exp
steps
local
data
local
lang
train
lang
Scripts and Files
-lexicon.txt
-nonsilence_phones.txt
-optional_silence.txt
-silence_phones.txt
-extra_questions.txt
Scripts and Files
-L_disambig.fst
-L.fst
-oov.int
-oov.txt
-phones
-phones.txt
-topo
-words.txt
Scripts and Files
-cmvn.scp
-feats.scp
-wav.scp
-segments
-text
-words.txt
-spk2utt
-utt2spk
conf
data
local
exp
• prepare_dict.sh
• prepare_data.sh
local
lang
lang_test
train/test/dev
lang
(tmp)
lm
dict
train
• lexicon.txt
• test
• text
• wav.scp
• utt2spk

Data Folder

- data/

File	Content
train/	Training set
test/	Test Set
dev/	Development set
dict/	Dictionary
graph/	Language model

  - train/, test/, dev/

File	Content
wav.scp	Waveform paths
text	Transcriptions
utt2spk	Utterances vs. Speakers
spk2utt	Speakers vs. Utterances

train/, test/, dev/

wav.scp

Speaker0001-0	~/kaldi-data/OC16-CE80/Training_Set/train/Speaker0001/0.wav
Speaker0001-1	~/kaldi-data/OC16-CE80/Training_Set/train/Speaker0001/1.wav
Speaker0001-10	~/kaldi-data/OC16-CE80/Training_Set/train/Speaker0001/10.wav

text

Speaker0001-0	打开 Notepad 编辑 文件
Speaker0001-1	结束 teleconference
Speaker0001-10	Capital Hotel 你 觉 的 怎么样

utt2spk

Speaker0001-0	Speaker0001
Speaker0001-1	Speaker0001
Speaker0001-10	Speaker0001

spk2utt

Speaker0001	Speaker0001-0 Speaker0001-1 Speaker0001-10 Speaker0001-11 … … …
Speaker0003	Speaker0003-1 Speaker0003-10 Speaker0003-11 Speaker0003-115 … … …
Speaker0004	Speaker0004-0 Speaker0004-1 Speaker0004-10 Speaker0004-11 … … …

Dictionary

- data/dict/

File	Content
lexicon.txt	dictionary
nonsilence_phones.txt	Phoneme set
silence_phones.txt	Silence
extra_questions.txt	Question set
optional_silence.txt	optional Silence

lexicon.txt

```
# sil
<SPOKEN_NOISE>      sil
<UNK>               sil
SIL                 sil
X光                 EH1 K S g uang1
X光线               EH1 K S g uang1 x
ian4
T恤                 T x v4
```

nonsilence_phones.txt

```
a1
a2
a3
a4
a5
aa
```

silence_phones.txt

```
sil
```

extra_questions.txt

```
sil
```

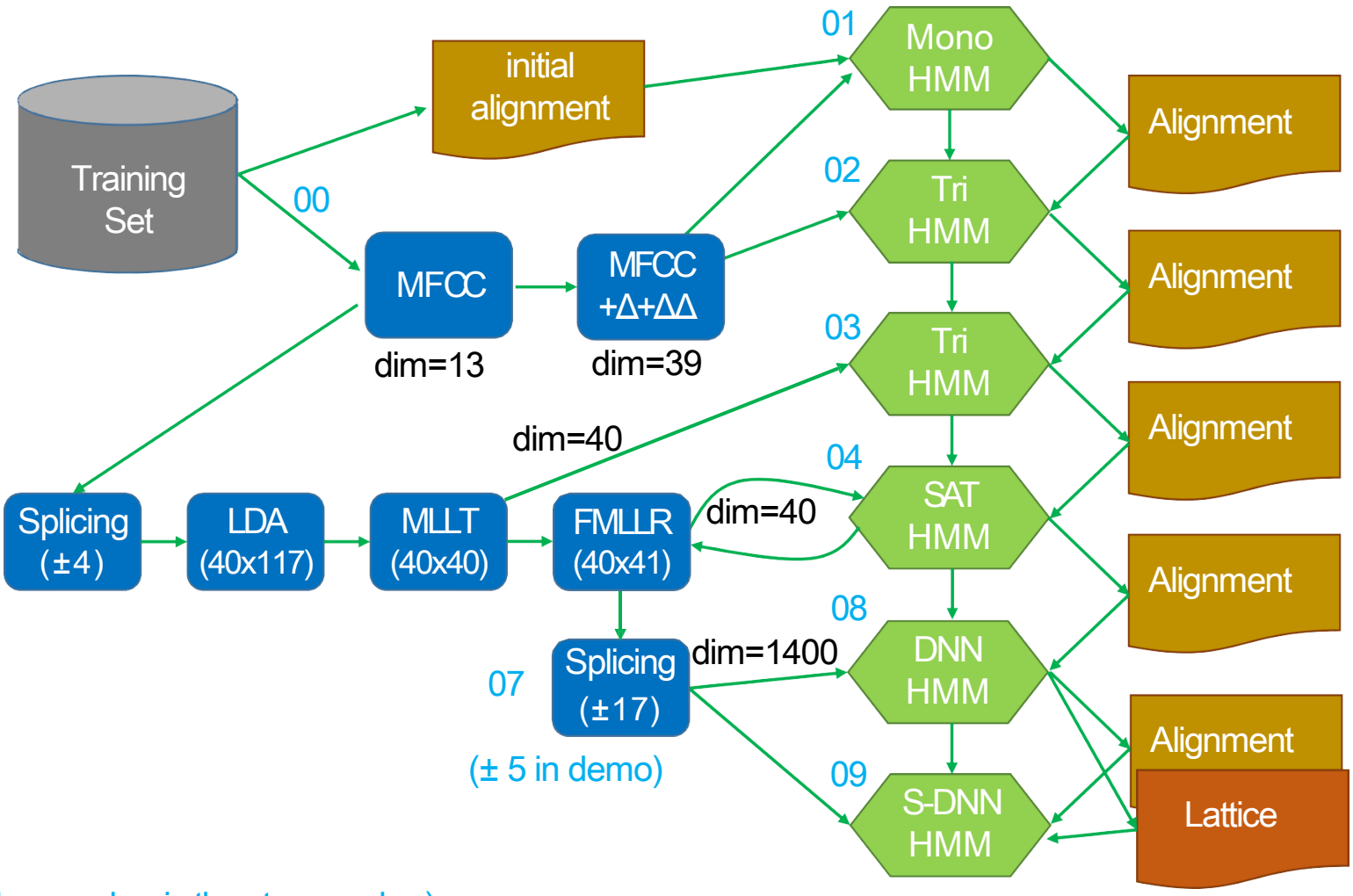
optional_silence.txt

```
sil
```

Building an STT System with Kaldi

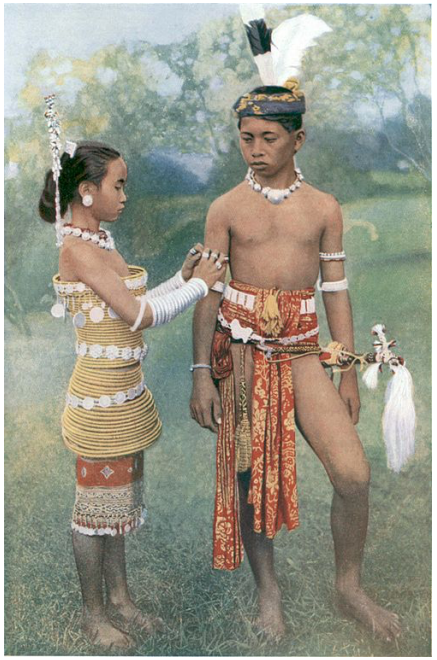
- Data preparation
  - Acoustic model training data
  - Pronunciation lexicon
  - Language model training data
- Basic GMM system building
  - Acoustic model training
  - Language model training
- Basic Decoding
  - Creating a static decoding graph
  - Lattice rescoring
- Basic DNN system building

Typical Training Procedure



(Blue number is the step number)

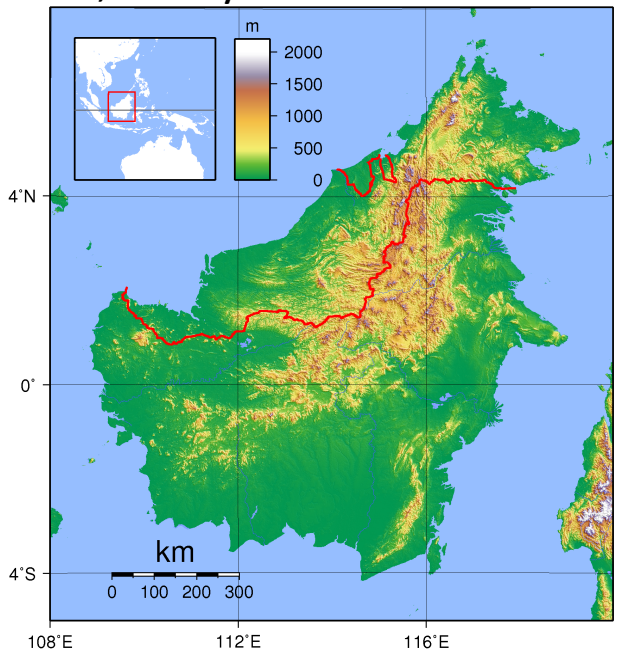
Kaldi-Recipe-Example



- Iban Corpus

Iban
Identifier: SLR24
Summary: Iban language text and speech corpora for ASR
Category: Speech
License: Attribution-ShareAlike 2.0 Generic (CC BY-SA 2.0)
Download: iban.tar.gz [913M] (Iban language corpora)

  - News data provided by a local radio station in Sarawak, Malaysia
  - Speech
    - Train 7 hours
    - Test 1 hour
  - Lexicon
    - 36350 words
  - LM
    - 2 M words



- Recipe
  - ~kaldi/egs/iban
    - ./run.sh

Setting up Paths, Queue Commands, …

- egs/iban/s5/run.sh

Terminal

```
# initialization PATH
. ./path.sh  || die "path.sh expected";
# initialization commands
. ./cmd.sh
. ./utils/parse_options.sh

set -e -o pipefail
corpus=./corpus
# download iban to build ASR
if [ ! -f "$corpus/README" ]; then
    #available from github
    mkdir -p ./$corpus/
    [ ! -f ./iban.tar.gz ] &&  wget http://www.openslr.org/resources/24/iban.tar.gz
    tar xzf iban.tar.gz -C $corpus
fi

nj=16
dev_nj=6

if [ $stage -le 1 ]; then
  echo "Preparing data and training language models"
  local/prepare_data.sh $corpus/
  local/prepare_dict.sh $corpus/
  utils/prepare_lang.sh data/local/dict "<UNK>" data/local/lang data/lang
  local/prepare_lm.sh
fi


if [ $stage -le 2 ]; then
  # Feature extraction
  for x in train dev; do
      steps/make_mfcc.sh --nj $nj --cmd "$train_cmd" data/$x exp/make_mfcc/$x mfcc
      steps/compute_cmvn_stats.sh data/$x exp/make_mfcc/$x mfcc
  done
fi
```

```
-uu-:---F1  run.sh          5% L46    (Shell-script[bash])-----------------------------------
```

Setting up Paths, Queue Commands, …

Terminal

```
export KALDI_ROOT=`pwd`/../../..
[ -f $KALDI_ROOT/tools/env.sh ] && . $KALDI_ROOT/tools/env.sh
export PATH=$PWD/utils/:$KALDI_ROOT/tools/openfst/bin:$PWD:$PATH
[ ! -f $KALDI_ROOT/tools/config/common_path.sh ] && echo >&2 "The standard file\
 $KALDI_ROOT/tools/config/common_path.sh is not present -> Exit!" && exit 1
. $KALDI_ROOT/tools/config/common_path.sh
export LC_ALL=C
```

Terminal

```
export train_cmd="run.pl --max-jobs-run 32"
export decode_cmd="run.pl --max-jobs-run 32"

#export train_cmd="queue.pl"
#export decode_cmd="queue.pl --mem 4G"

export train_cmd="run.pl"
export decode_cmd="run.pl --mem 4G"
```

Building an STT System with Kaldi

- Data preparation
  - **Acoustic model training data**
  - Pronunciation lexicon
  - Language model training data
- Basic GMM system building
  - Acoustic model training
  - Language model training
- Basic Decoding
  - Creating a static decoding graph
  - Lattice rescoring
- Basic DNN system building
- Going beyond the basics

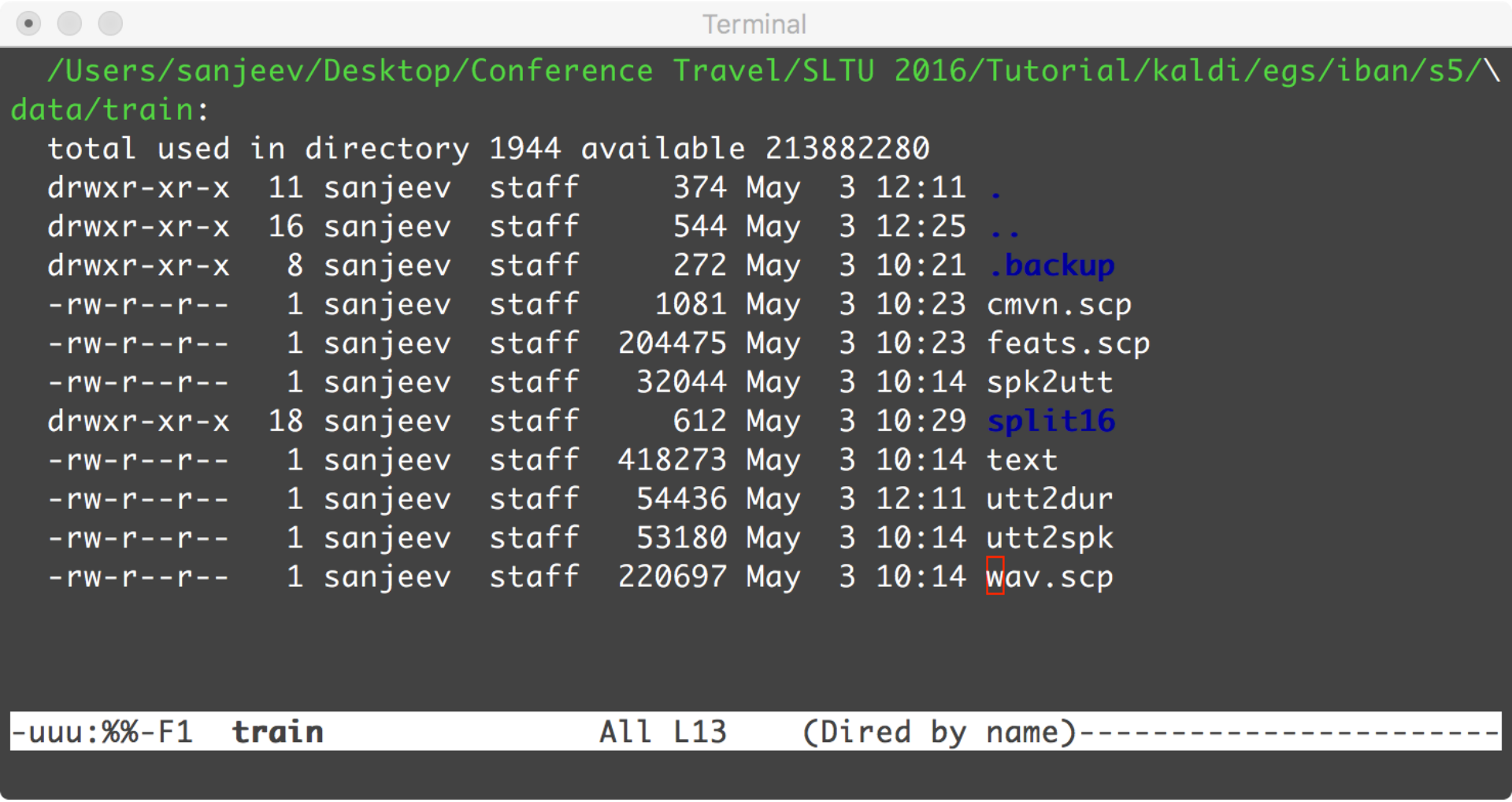
Preparing Acoustic Training Data

Terminal

```
# initialization PATH
. ./path.sh  || die "path.sh expected";
# initialization commands
. ./cmd.sh
. ./utils/parse_options.sh

set -e -o pipefail
corpus=./corpus
# download iban to build ASR
if [ ! -f "$corpus/README" ]; then
    #available from github
    mkdir -p ./$corpus/
    [ ! -f ./iban.tar.gz ] &&  wget http://www.openslr.org/resources/24/iban.tar.gz
    tar xzf iban.tar.gz -C $corpus
fi

nj=16
dev_nj=6

if [ $stage -le 1 ]; then
  echo "Preparing data and training language models"
  local/prepare_data.sh $corpus/
  local/prepare_dict.sh $corpus/
  utils/prepare_lang.sh data/local/dict "<UNK>" data/local/lang data/lang
  local/prepare_lm.sh
fi


if [ $stage -le 2 ]; then
  # Feature extraction
  for x in train dev; do
      steps/make_mfcc.sh --nj $nj --cmd "$train_cmd" data/$x exp/make_mfcc/$x mfcc
      steps/compute_cmvn_stats.sh data/$x exp/make_mfcc/$x mfcc
  done
fi
```

-uu-:---F1 **run.sh** 5% L46 (Shell-script[bash])----------

Preparing Acoustic Training Data

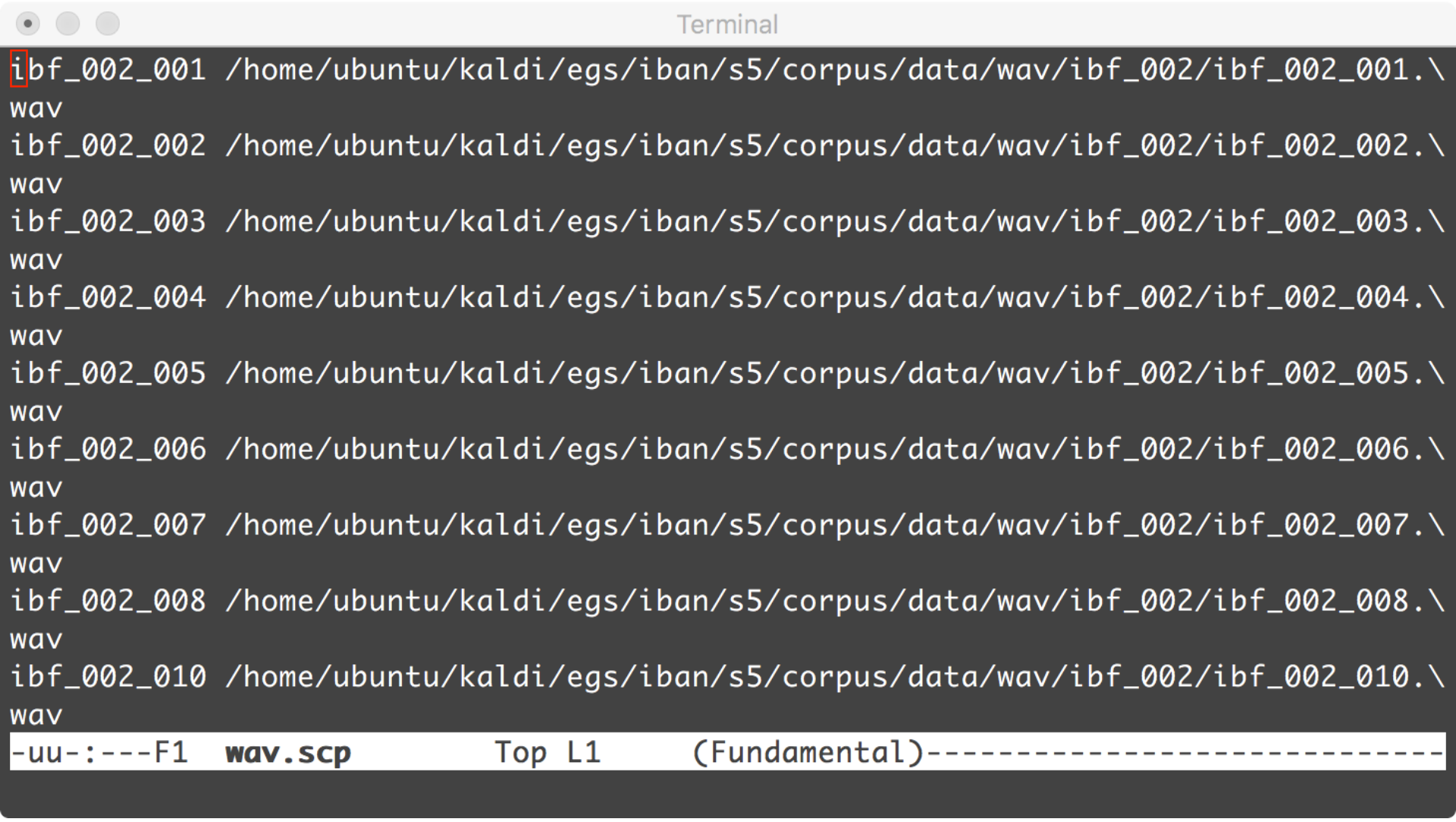


data/train/text

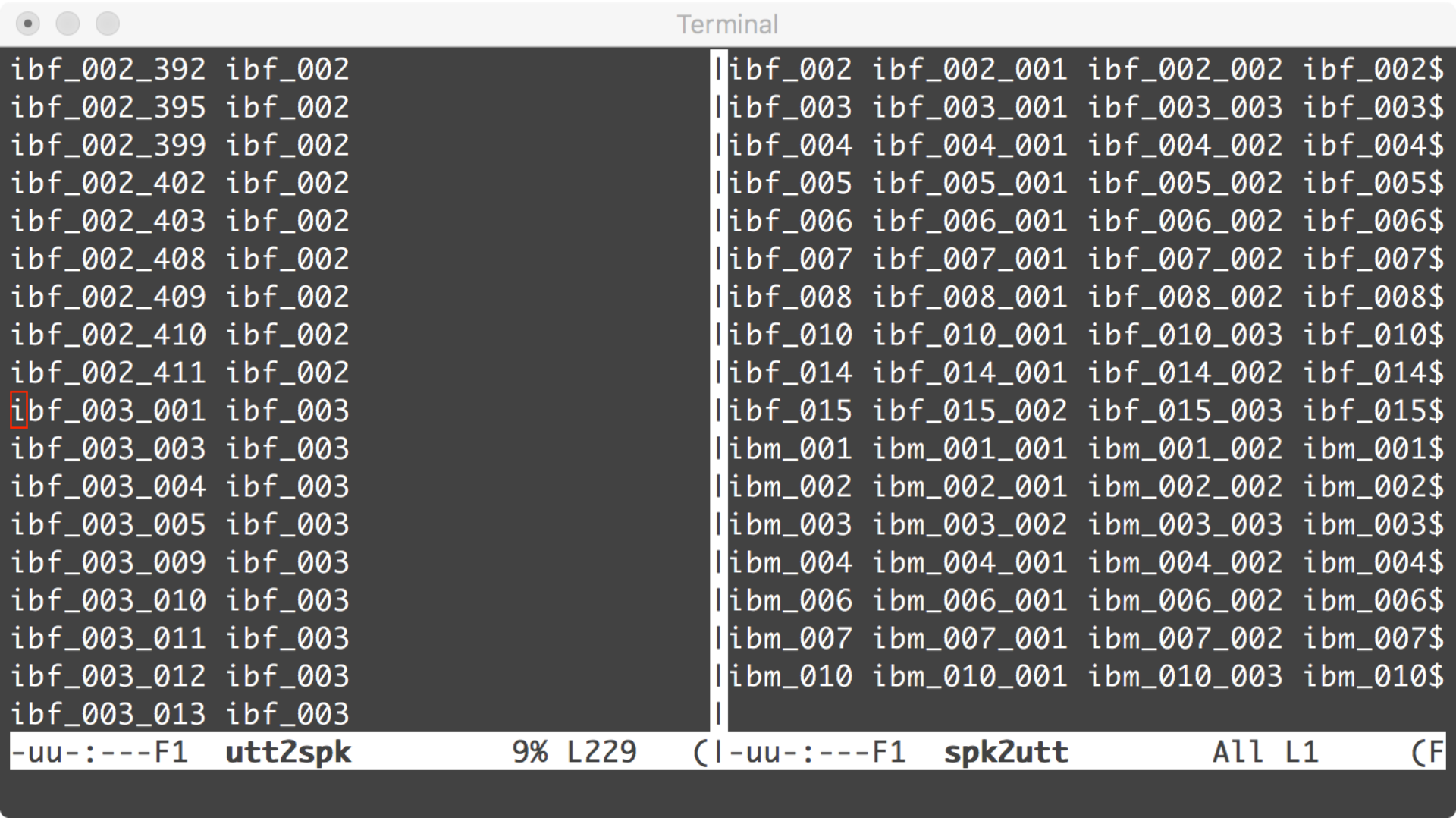
Terminal

```
ibf_002_165  bagiiban miri fm bisi nyediaka tempat ke empat bengkah program gaw\
ai rambau musim pengerami gawai ke taun tu
ibf_002_166  program gawai tu nyengkaum jaku pesan ari penulung menteri perengk\
a ke diguna mensia mayuh datuk sylvester entri muran enggau
ibf_002_167  program anak mit begawai ba studio ke dikeluarka mary bajang
ibf_002_169  kepala bagi iban rtm miri encik simon suti madahka bala nembiak ke\
 masuk program setengah jam nya datai ari sk lambir village sk beluru central e\
nggau sk kelapa sawit numor dua
ibf_002_171  program nya deka ditabur ba serata menua nengah waifm ba ari ti ke\
terubah gawai pukul sepuluh pagi
ibf_002_172  sebengkah agi program gawai nya berami gawai miri fm dua ribu dua \
belas pengelama empat puluh lima minit
ibf_002_174  program nya deka dikeluar ba ari ti kedua gawai pukul dua ngalih h\
ari
ibf_002_175  nya naka kami naburka berita ari waifm berita nya tadi ditusun bev\
erly kaur sereta disalin shirley limban salam satu malaysia
ibf_002_177  pukul tujuh pagi
ibf_002_179  diatu kami naburka berita ari waifm
-uu-:---F1  text                4% L89    (Fundamental)----------------------------
Undo!
```

data/train/wav.scp



data/train/(utt2spk|spk2utt)



data/train/(cmvn.scp|feats.scp)

Terminal

```
# initialization PATH
. ./path.sh  || die "path.sh expected";
# initialization commands
. ./cmd.sh
. ./utils/parse_options.sh

set -e -o pipefail
corpus=./corpus
# download iban to build ASR
if [ ! -f "$corpus/README" ]; then
    #available from github
    mkdir -p ./$corpus/
    [ ! -f ./iban.tar.gz ] &&  wget http://www.openslr.org/resources/24/iban.tar.gz
    tar xzf iban.tar.gz -C $corpus
fi

nj=16
dev_nj=6

if [ $stage -le 1 ]; then
  echo "Preparing data and training language models"
  local/prepare_data.sh $corpus/
  local/prepare_dict.sh $corpus/
  utils/prepare_lang.sh data/local/dict "<UNK>" data/local/lang data/lang
  local/prepare_lm.sh
fi


if [ $stage -le 2 ]; then
  # Feature extraction
  for x in train dev; do
      steps/make_mfcc.sh --nj $nj --cmd "$train_cmd" data/$x exp/make_mfcc/$x mfcc
      steps/compute_cmvn_stats.sh data/$x exp/make_mfcc/$x mfcc
  done
fi
```

-uu-:---F1 **run.sh** 5% L46 (Shell-script[bash])----------

data/train/(cmvn.scp|feats.scp)

Terminal

```
ibf_002 /home/ubuntu/kaldi/egs/iban/s5/mfcc/cmvn_train.ark:8
ibf_003 /home/ubuntu/kaldi/egs/iban/s5/mfcc/cmvn_train.ark:255
ibf_004 /home/ubuntu/kaldi/egs/iban/s5/mfcc/cmvn_train.ark:502
ibf_005 /home/ubuntu/kaldi/egs/iban/s5/mfcc/cmvn_train.ark:749
ibf_006 /home/ubuntu/kaldi/egs/iban/s5/mfcc/cmvn_train.ark:996
ibf_007 /home/ubuntu/kaldi/egs/iban/s5/mfcc/cmvn_train.ark:1243
ibf_008 /home/ubuntu/kaldi/egs/iban/s5/mfcc/cmvn_train.ark:1490
ibf_010 /home/ubuntu/kaldi/egs/iban/s5/mfcc/cmvn_train.ark:1737
-uu-:---F1  cmvn.scp        Top L1     (Fundamental)----------------------------
ibf_002_001 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:12
ibf_002_002 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:13877
ibf_002_003 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:32877
ibf_002_004 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:38292
ibf_002_005 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:56876
ibf_002_006 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:71261
ibf_002_007 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:84346
ibf_002_008 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:104425
ibf_002_010 /home/ubuntu/kaldi/egs/iban/s5/mfcc/raw_mfcc_train.1.ark:121202
-uu-:---F1  feats.scp        Top L1     (Fundamental)----------------------------
```

Building an STT System with Kaldi

- Data preparation
  - Acoustic model training data
  - **Pronunciation lexicon**
  - Language model training data
- Basic GMM system building
  - Acoustic model training
  - Language model training
- Basic Decoding
  - Creating a static decoding graph
  - Lattice rescoring
- Basic DNN system building
- Going beyond the basics

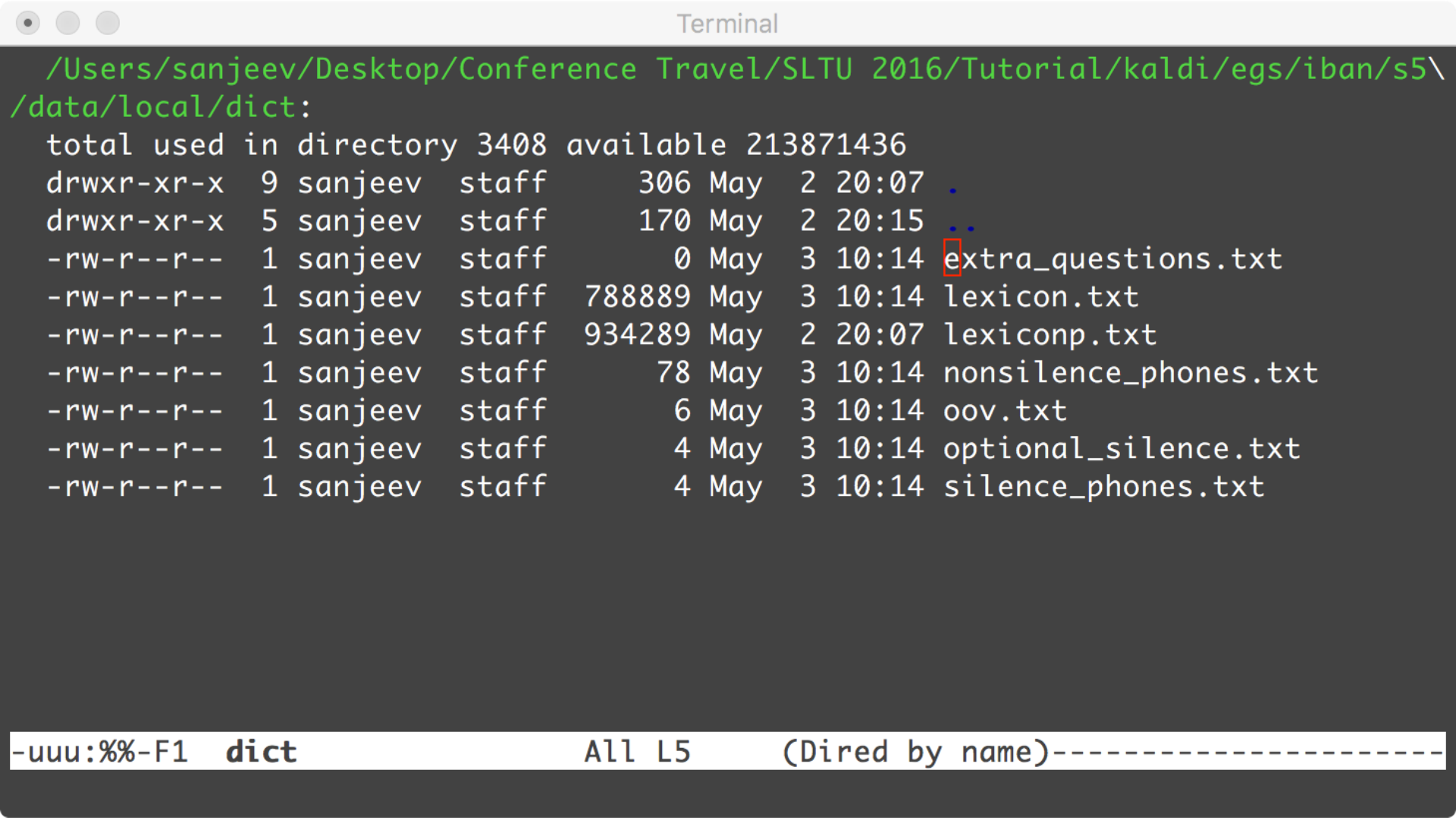
Preparing the Pronunciation Lexicon

Terminal

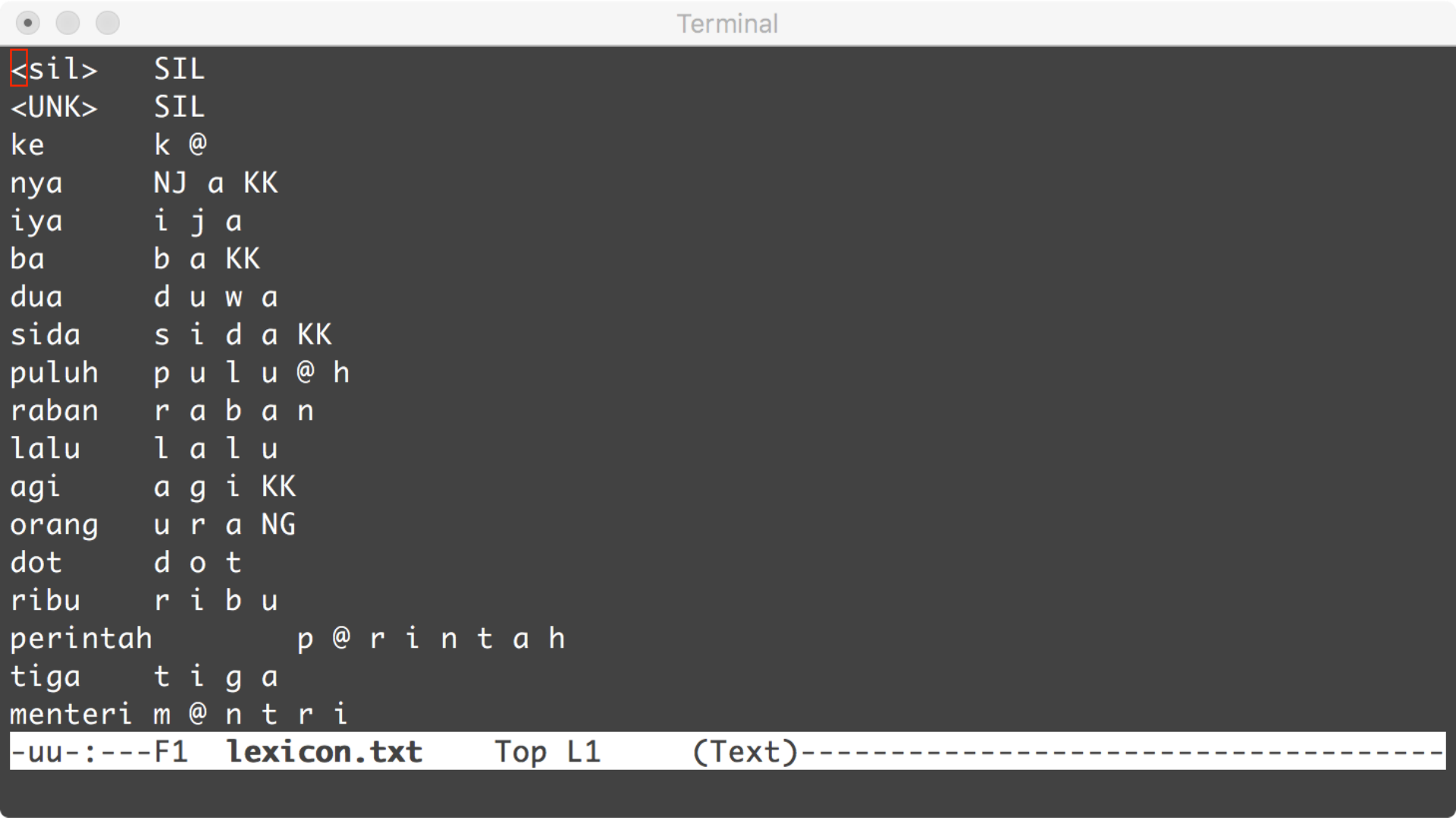
```
# initialization PATH
. ./path.sh  || die "path.sh expected";
# initialization commands
. ./cmd.sh
. ./utils/parse_options.sh

set -e -o pipefail
corpus=./corpus
# download iban to build ASR
if [ ! -f "$corpus/README" ]; then
    #available from github
    mkdir -p ./$corpus/
    [ ! -f ./iban.tar.gz ] &&  wget http://www.openslr.org/resources/24/iban.tar.gz
    tar xzf iban.tar.gz -C $corpus
fi

nj=16
dev_nj=6

if [ $stage -le 1 ]; then
  echo "Preparing data and training language models"
  local/prepare_data.sh $corpus/
  local/prepare_dict.sh $corpus/
  utils/prepare_lang.sh data/local/dict "<UNK>" data/local/lang data/lang
  local/prepare_lm.sh
fi


if [ $stage -le 2 ]; then
  # Feature extraction
  for x in train dev; do
      steps/make_mfcc.sh --nj $nj --cmd "$train_cmd" data/$x exp/make_mfcc/$x mfcc
      steps/compute_cmvn_stats.sh data/$x exp/make_mfcc/$x mfcc
  done
fi
```

```
-uu-:---F1  run.sh          5% L46    (Shell-script[bash])-----------------------------------
```

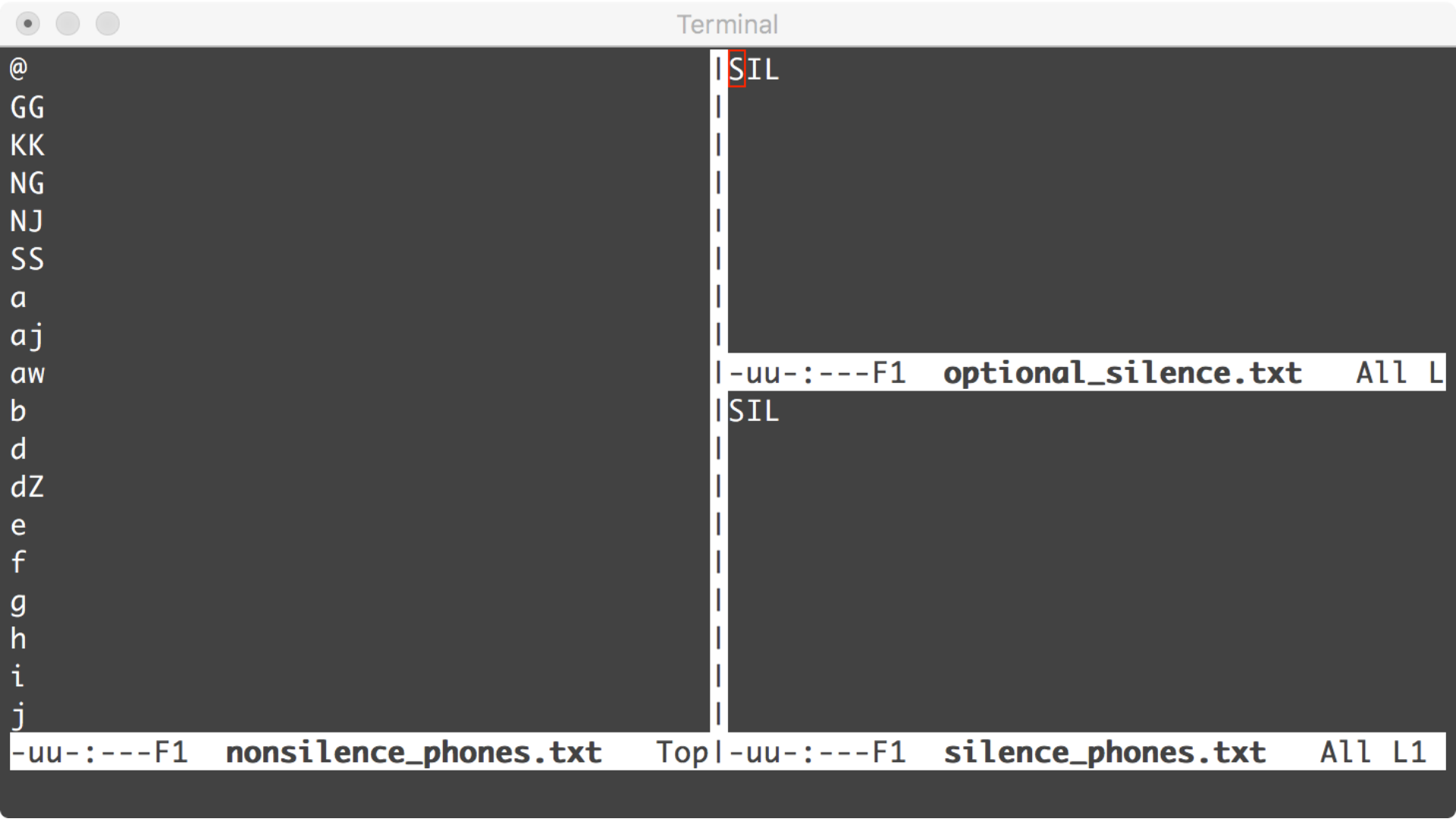
Preparing the Pronunciation Lexicon



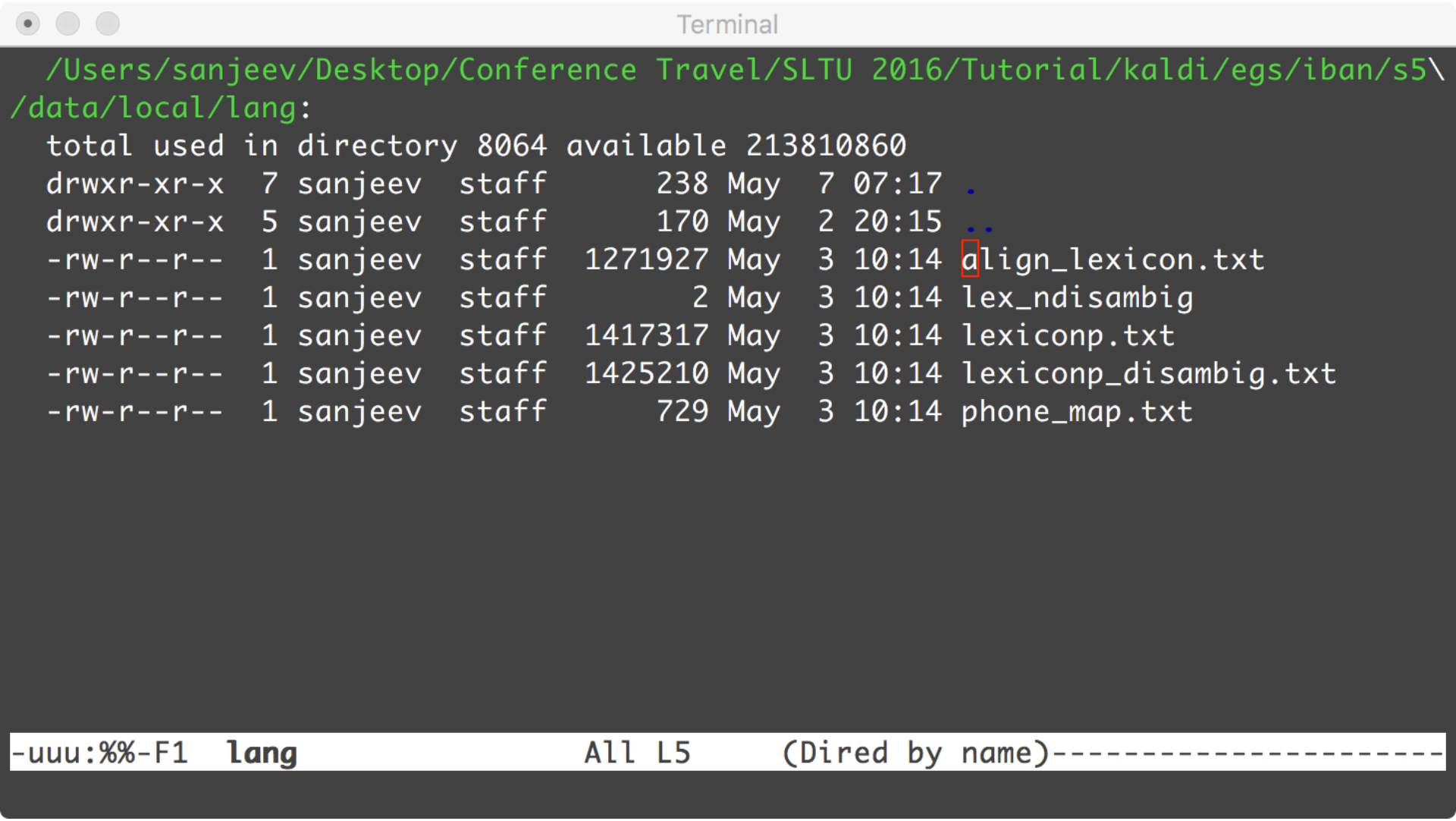
data/local/dict/lexicon.txt



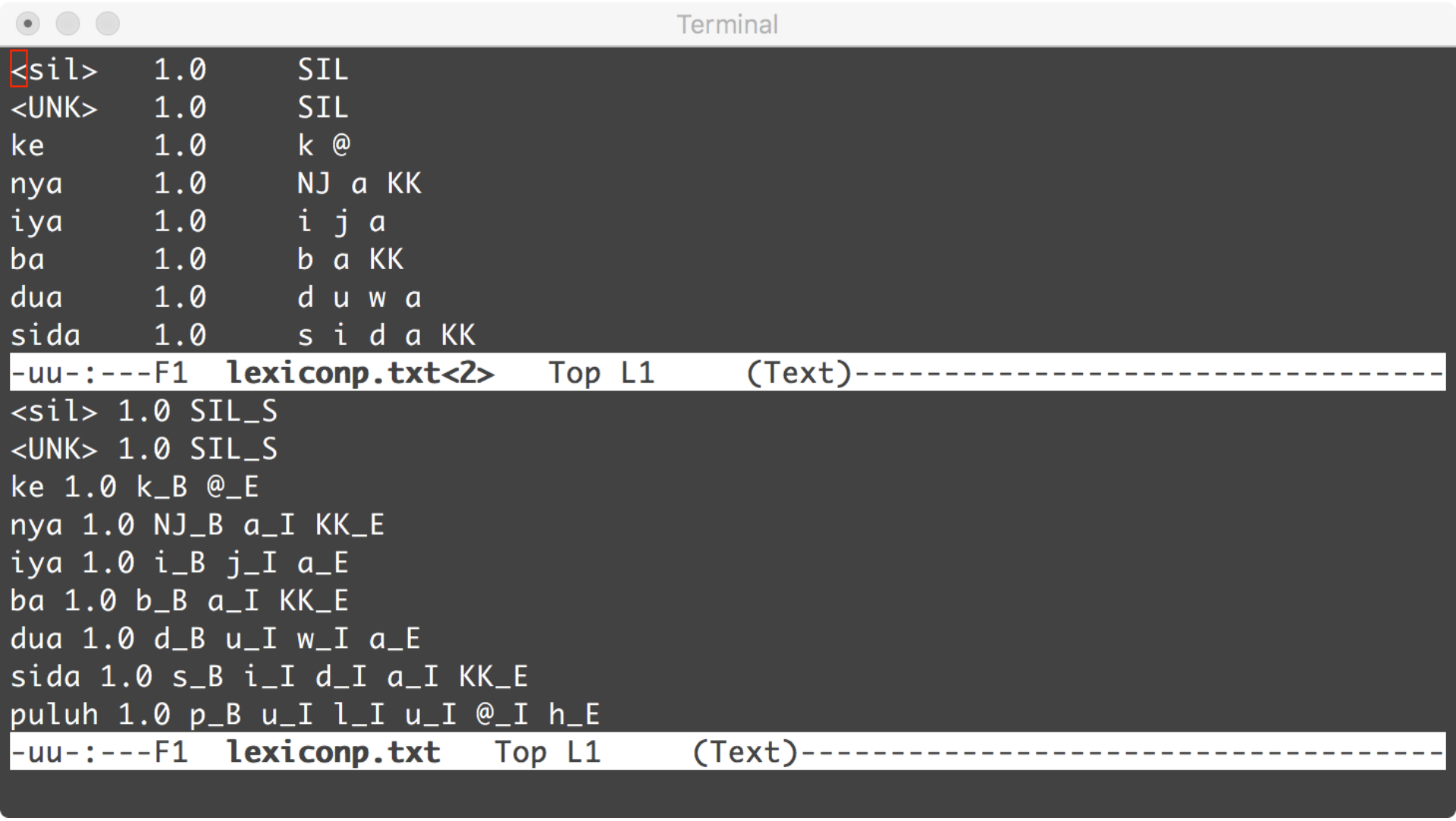
data/local/dict/*silence*.txt



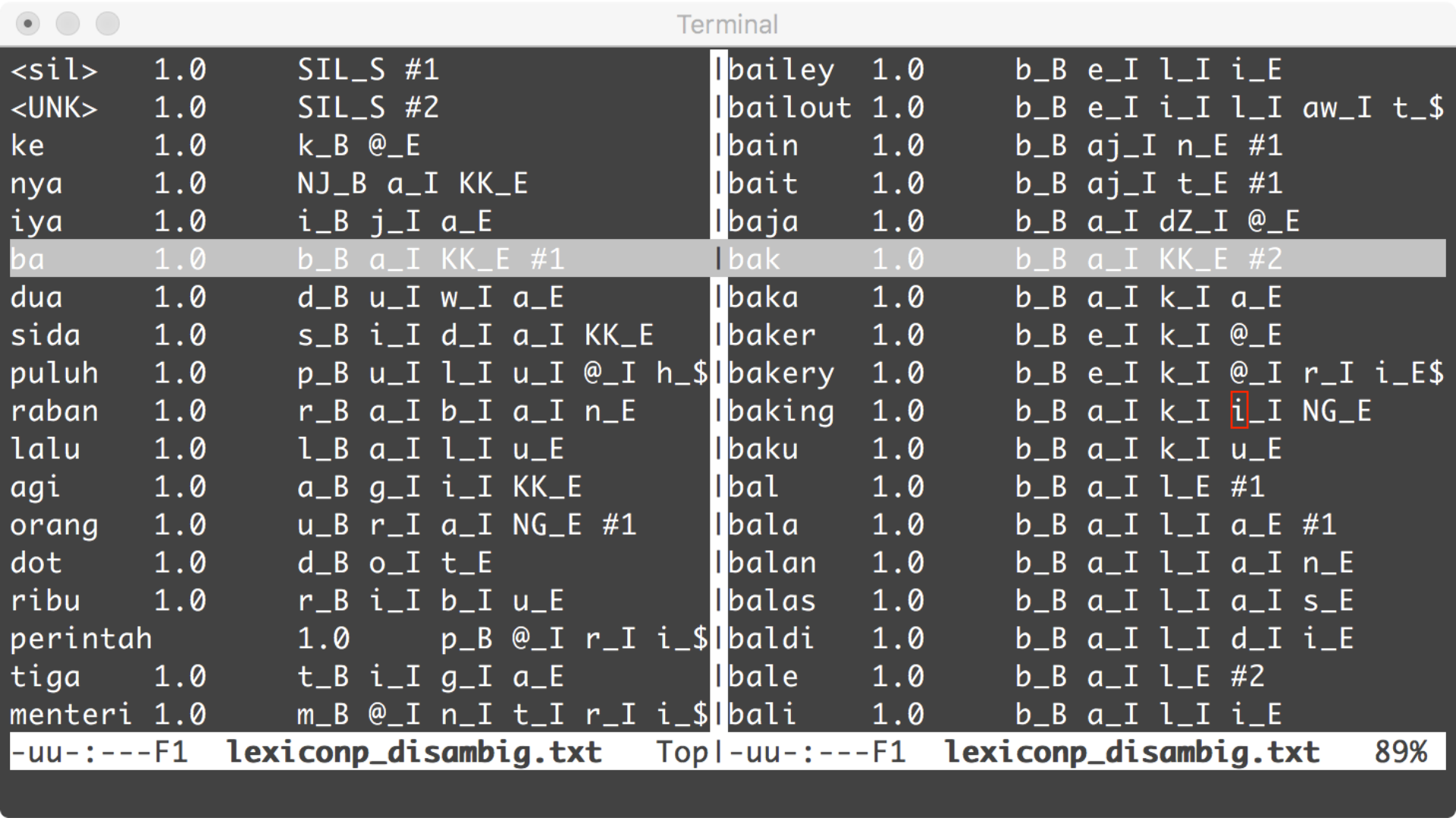
data/local/lang



Word Boundary Tags



Disambiguation Symbols

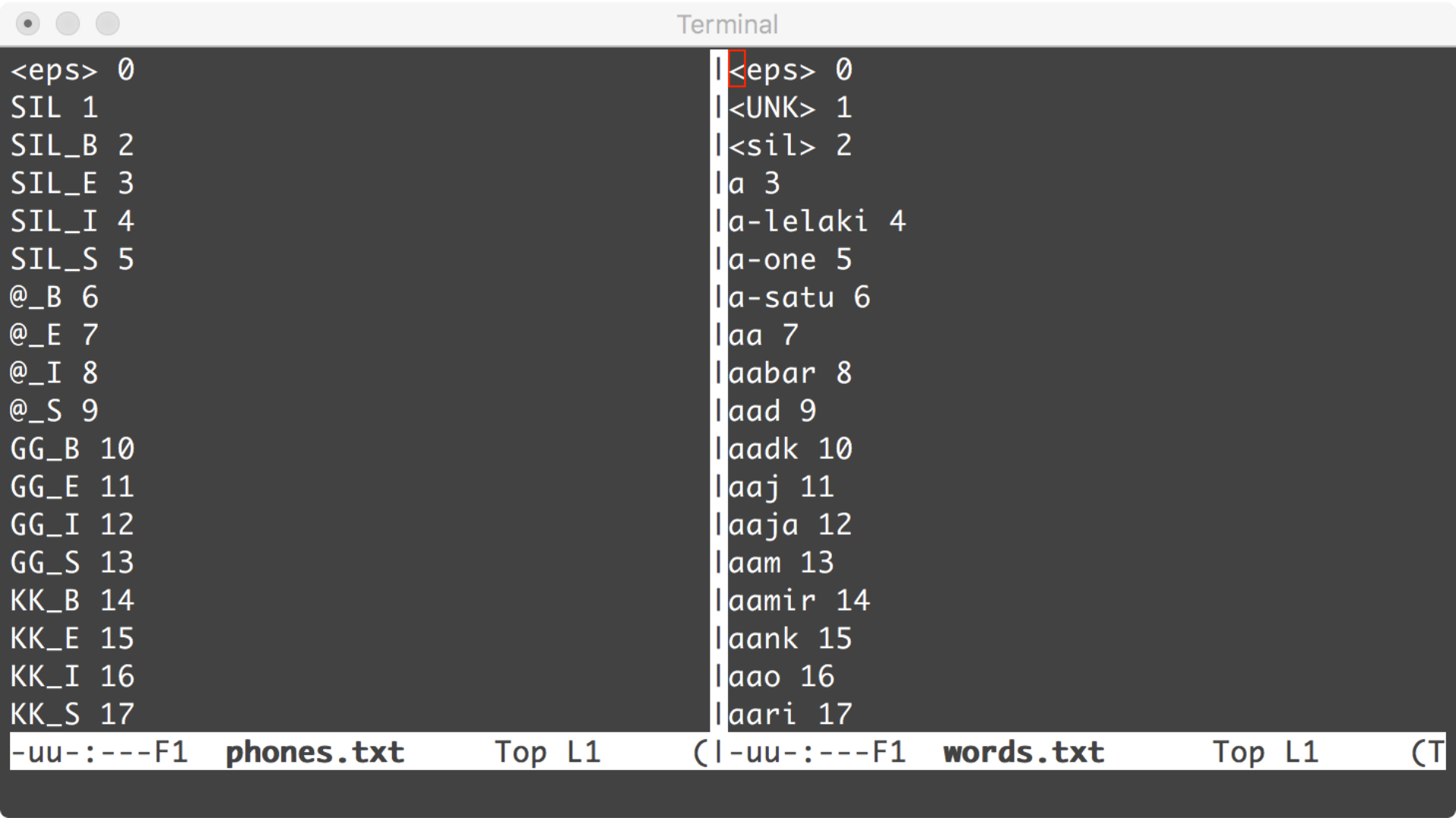


data/lang

Terminal

```
  /Users/sanjeev/Desktop/Conference Travel/SLTU 2016/Tutorial/kaldi/egs/iban/s5\
/data/lang:
  total used in directory 28128 available 213813416
  drwxr-xr-x  10 sanjeev  staff      340 May  2 20:07 .
  drwxr-xr-x  16 sanjeev  staff      544 May  3 12:25 ..
  -rw-r--r--   1 sanjeev  staff  6908222 May  3 10:14 L.fst
  -rw-r--r--   1 sanjeev  staff  6981934 May  3 10:14 L_disambig.fst
  -rw-r--r--   1 sanjeev  staff        2 May  3 10:14 oov.int
  -rw-r--r--   1 sanjeev  staff        6 May  3 10:14 oov.txt
  drwxr-xr-x  30 sanjeev  staff     1020 May  2 20:07 phones
  -rw-r--r--   1 sanjeev  staff     1146 May  3 10:14 phones.txt
  -rw-r--r--   1 sanjeev  staff     1369 May  3 10:14 topo
  -rw-r--r--   1 sanjeev  staff   490107 May  3 10:14 words.txt

-uuu:%%-F1  lang<2>            All L10    (Dired by name)----------------------
```

data/lang/(phones|words).txt

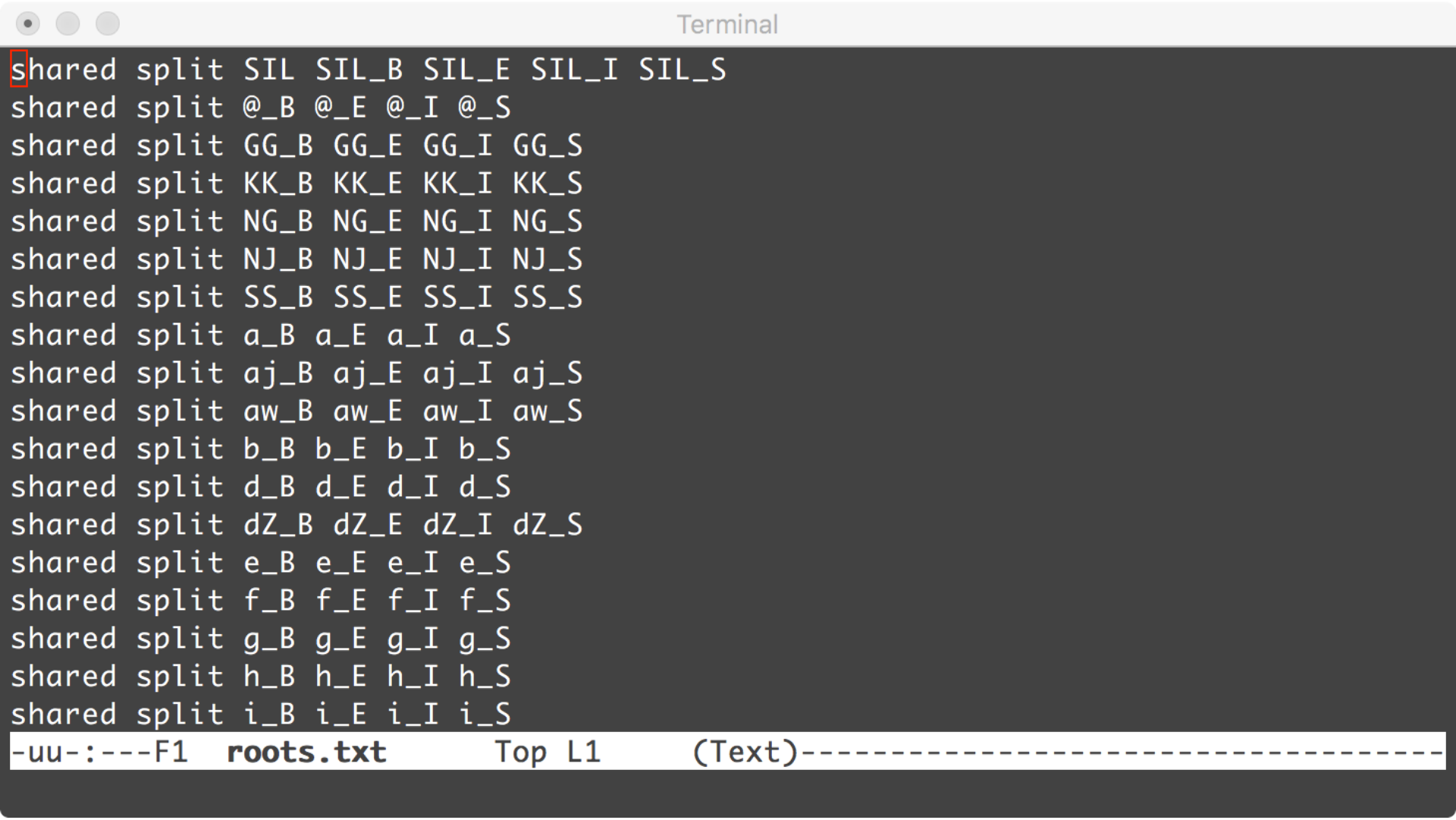


data/lang/topo

Terminal

```
<ForPhones>
1 2 3 4 5
</ForPhones>
<State> 0 <PdfClass> 0 <Transition> 0 0.25 <Transition> 1 0.25 <Transition> 2 0\
.25 <Transition> 3 0.25 </State>
<State> 1 <PdfClass> 1 <Transition> 1 0.25 <Transition> 2 0.25 <Transition> 3 0\
.25 <Transition> 4 0.25 </State>
<State> 2 <PdfClass> 2 <Transition> 1 0.25 <Transition> 2 0.25 <Transition> 3 0\
.25 <Transition> 4 0.25 </State>
<State> 3 <PdfClass> 3 <Transition> 1 0.25 <Transition> 2 0.25 <Transition> 3 0\
.25 <Transition> 4 0.25 </State>
<State> 4 <PdfClass> 4 <Transition> 4 0.75 <Transition> 5 0.25 </State>
<State> 5 </State>
</TopologyEntry>
</Topology>


-uu-:---F1  topo          Bot L12    (Fundamental)------------------------------
```

data/lang/phones/roots.txt



data/lang/phones/extra_questions.txt

Terminal

```
@_B GG_B KK_B NG_B NJ_B SS_B a_B aj_B aw_B b_B d_B dZ_B e_B f_B g_B h_B i_B j_B\
 k_B l_B m_B n_B o_B oj_B p_B r_B s_B t_B tS_B u_B v_B w_B x_B z_B
@_E GG_E KK_E NG_E NJ_E SS_E a_E aj_E aw_E b_E d_E dZ_E e_E f_E g_E h_E i_E j_E\
 k_E l_E m_E n_E o_E oj_E p_E r_E s_E t_E tS_E u_E v_E w_E x_E z_E
@_I GG_I KK_I NG_I NJ_I SS_I a_I aj_I aw_I b_I d_I dZ_I e_I f_I g_I h_I i_I j_I\
 k_I l_I m_I n_I o_I oj_I p_I r_I s_I t_I tS_I u_I v_I w_I x_I z_I
@_S GG_S KK_S NG_S NJ_S SS_S a_S aj_S aw_S b_S d_S dZ_S e_S f_S g_S h_S i_S j_S\
 k_S l_S m_S n_S o_S oj_S p_S r_S s_S t_S tS_S u_S v_S w_S x_S z_S
SIL
SIL_B
SIL_E
SIL_I
SIL_S
```

-uu-:---F1 **extra_questions.txt**   All L1   (Text)----------

Building an STT System with Kaldi

- Data preparation
  - Acoustic model training data
  - Pronunciation lexicon
  - **Language model training data**
- Basic GMM system building
  - Acoustic model training
  - Language model training
- Basic Decoding
  - Creating a static decoding graph
  - Lattice rescoring
- Basic DNN system building
- Going beyond the basics

Preparing the Language Model

Terminal

```
# initialization PATH
. ./path.sh  || die "path.sh expected";
# initialization commands
. ./cmd.sh
. ./utils/parse_options.sh

set -e -o pipefail
corpus=./corpus
# download iban to build ASR
if [ ! -f "$corpus/README" ]; then
    #available from github
    mkdir -p ./$corpus/
    [ ! -f ./iban.tar.gz ] &&  wget http://www.openslr.org/resources/24/iban.tar.gz
    tar xzf iban.tar.gz -C $corpus
fi

nj=16
dev_nj=6

if [ $stage -le 1 ]; then
  echo "Preparing data and training language models"
  local/prepare_data.sh $corpus/
  local/prepare_dict.sh $corpus/
  utils/prepare_lang.sh data/local/dict "<UNK>" data/local/lang data/lang
  local/prepare_lm.sh
fi


if [ $stage -le 2 ]; then
  # Feature extraction
  for x in train dev; do
      steps/make_mfcc.sh --nj $nj --cmd "$train_cmd" data/$x exp/make_mfcc/$x mfcc
      steps/compute_cmvn_stats.sh data/$x exp/make_mfcc/$x mfcc
  done
fi
```

-uu-:---F1 **run.sh** 5% L46 (Shell-script[bash])----------

Preparing the Language Model

Terminal

```
local/train_lms_srilm.sh --train-text data/train/text data/ data/srilm

nl -nrz -w10  corpus/LM/iban-bp-2012.txt | sort -R > data/local/external_text
local/train_lms_srilm.sh --train-text data/local/external_text data/ data/srilm\
_external

# let's do ngram interpolation of the previous two LMs
# the lm.gz is always symlink to the model with the best perplexity, so we use \
that

mkdir -p data/srilm_interp
for w in 0.9 0.8 0.7 0.6 0.5; do
    ngram -lm data/srilm/lm.gz  -mix-lm data/srilm_external/lm.gz \
          -lambda $w -write-lm data/srilm_interp/lm.${w}.gz
    echo -n "data/srilm_interp/lm.${w}.gz "
    ngram -lm data/srilm_interp/lm.${w}.gz -ppl data/srilm/dev.txt | paste -s
done | sort  -k15,15g  > data/srilm_interp/perplexities.txt
-uu-:---F1  prepare_lm.sh   14% L19    (Shell-script[bash])--------------------
```

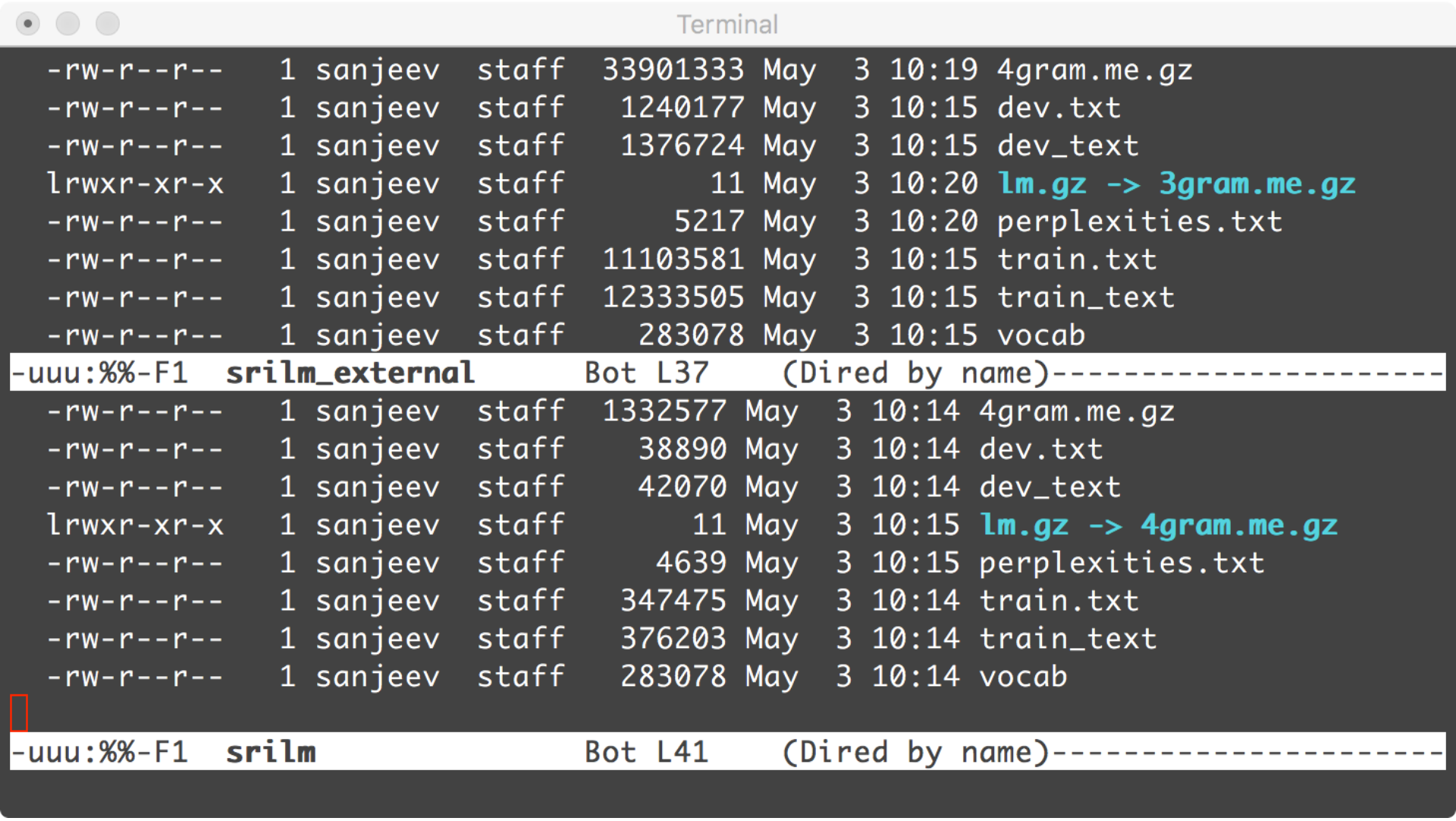
local/train_lms_srilm.sh

Terminal

```
echo "-------------------"
echo "Kneser-Ney 3grams"
echo "-------------------"
ngram-count -lm $tgtdir/3gram.kn011.gz -kndiscount1 -gt1min 0 -kndiscount2 -gt2\
min 1 -kndiscount3 -gt3min 1 -order 3 -text $tgtdir/train.txt -vocab $tgtdir/vo\
cab -unk -sort -map-unk "$oov_symbol"
ngram-count -lm $tgtdir/3gram.kn012.gz -kndiscount1 -gt1min 0 -kndiscount2 -gt2\
min 1 -kndiscount3 -gt3min 2 -order 3 -text $tgtdir/train.txt -vocab $tgtdir/vo\
cab -unk -sort -map-unk "$oov_symbol"
ngram-count -lm $tgtdir/3gram.kn022.gz -kndiscount1 -gt1min 0 -kndiscount2 -gt2\
min 2 -kndiscount3 -gt3min 2 -order 3 -text $tgtdir/train.txt -vocab $tgtdir/vo\
cab -unk -sort -map-unk "$oov_symbol"
ngram-count -lm $tgtdir/3gram.kn023.gz -kndiscount1 -gt1min 0 -kndiscount2 -gt2\
min 2 -kndiscount3 -gt3min 3 -order 3 -text $tgtdir/train.txt -vocab $tgtdir/vo\
cab -unk -sort -map-unk "$oov_symbol"
```

-uu-:---F1 **train_lms_srilm.sh** 48% L152 (Shell-script[bash])----------

local/train_lms_srilm.sh (cont'd)



Interpolated Language Models

Terminal

```
# let's do ngram interpolation of the previous two LMs
# the lm.gz is always symlink to the model with the best perplexity, so we use \
that

mkdir -p data/srilm_interp
for w in 0.9 0.8 0.7 0.6 0.5; do
    ngram -lm data/srilm/lm.gz  -mix-lm data/srilm_external/lm.gz \
          -lambda $w -write-lm data/srilm_interp/lm.${w}.gz
    echo -n "data/srilm_interp/lm.${w}.gz "
    ngram -lm data/srilm_interp/lm.${w}.gz -ppl data/srilm/dev.txt | paste -s
done | sort  -k15,15g  > data/srilm_interp/perplexities.txt

# for basic decoding, let's use only a trigram LM
[ -d data/lang_test/ ] && rm -rf data/lang_test
cp -R data/lang data/lang_test
lm=$(cat data/srilm/perplexities.txt | grep 3gram | head -n1 | awk '{print $1}'\
)
local/arpa2G.sh $lm data/lang_test data/lang_test
-uu-:---F1  prepare_lm.sh   29% L24     (Shell-script[bash])--------------------
```

local/arpa2G.sh

Terminal

```
$decompress | \
  grep -v '<s> <s>' | grep -v '</s> <s>' |  grep -v '</s> </s>' | \
  arpa2fst - | \
  fstprint | \
  utils/eps2disambig.pl | \
  utils/s2eps.pl | \
  fstcompile --isymbols=$langdir/words.txt \
  --osymbols=$langdir/words.txt  --keep_isymbols=false --keep_osymbols=false | \
\
 fstrmepsilon | fstarcsort --sort_type=olabel > $destdir/G.fst || exit 1
fstisstochastic $destdir/G.fst || true;

if $cleanup; then
  rm $destdir/lm_tmp.gz  2>/dev/null || true;
fi

exit 0

-uu-:---F1  arpa2G.sh       Bot L108    (Shell-script[bash])----------------------
```

Building an STT System with Kaldi

- Data preparation
  - Acoustic model training data
  - Pronunciation lexicon
  - Language model training data
- Basic GMM system building
  - **Acoustic model training**
  - Language model training
- Basic Decoding
  - Creating a static decoding graph
  - Lattice rescoring
- Basic DNN system building
- Going beyond the basics

GMM Training (1)

Terminal

```
if [ $stage -le 3 ]; then
  ### Monophone
  echo "Starting monophone training."
  utils/subset_data_dir.sh data/train 1000 data/train.1k
  steps/train_mono.sh --nj $nj --cmd "$train_cmd" data/train.1k data/lang exp/m\
ono
  echo "Mono training done."

  (
  echo "Decoding the dev set using monophone models."
  utils/mkgraph.sh --mono data/lang_test exp/mono exp/mono/graph

  steps/decode.sh --config conf/decode.config --nj $dev_nj --cmd "$decode_cmd" \
\
    exp/mono/graph data/dev exp/mono/decode_dev
  echo "Monophone decoding done."
  ) &
fi


if [ $stage -le 4 ]; then
  ### Triphone
  echo "Starting triphone training."
  steps/align_si.sh --nj $nj --cmd "$train_cmd" \
      data/train data/lang exp/mono exp/mono_ali
  steps/train_deltas.sh --boost-silence 1.25  --cmd "$train_cmd"  \
      3200 30000 data/train data/lang exp/mono_ali exp/tri1
  echo "Triphone training done."

  (
  echo "Decoding the dev set using triphone models."
  utils/mkgraph.sh data/lang_test  exp/tri1 exp/tri1/graph
  steps/decode.sh --nj $dev_nj --cmd "$decode_cmd"  \
      exp/tri1/graph  data/dev exp/tri1/decode_dev

  steps/lmrescore_const_arpa.sh  --cmd "$decode_cmd" \
      data/lang_test/ data/lang_big/ data/dev \
-uu-:---F1  run.sh         19% L64    (Shell-script[bash])----------------------
```

GMM Training (1)

Terminal

```
if [ $stage -le 3 ]; then
  ### Monophone
  echo "Starting monophone training."
  utils/subset_data_dir.sh data/train 1000 data/train.1k
  steps/train_mono.sh --nj $nj --cmd "$train_cmd" data/train.1k data/lang exp/m\
ono
  echo "Mono training done."

-uu-:---F1   run.sh          19% L52     (Shell-script[bash])----------------------
if [ $stage -le 4 ]; then
  ### Triphone
  echo "Starting triphone training."
  steps/align_si.sh --nj $nj --cmd "$train_cmd" \
      data/train data/lang exp/mono exp/mono_ali
  steps/train_deltas.sh --boost-silence 1.25  --cmd "$train_cmd"  \
      3200 30000 data/train data/lang exp/mono_ali exp/tri1
  echo "Triphone training done."

-uu-:---F1   run.sh          28% L67     (Shell-script[bash])----------------------
```

GMM Training (2)

Terminal

```
#!/bin/bash
# Copyright 2012  Johns Hopkins University (Author: Daniel Povey)
# Apache 2.0


# To be run from ..
# Flat start and monophone training, with delta-delta features.
# This script applies cepstral mean normalization (per speaker).

# Begin configuration section.
nj=4
cmd=run.pl
scale_opts="--transition-scale=1.0 --acoustic-scale=0.1 --self-loop-scale=0.1"
num_iters=40    # Number of iterations of training
max_iter_inc=30 # Last iter to increase #Gauss on.
totgauss=1000 # Target #Gaussians.
careful=false
boost_silence=1.0 # Factor by which to boost silence likelihoods in alignment
-uu-:---F1  train_mono.sh   Top L1     (Shell-script[bash])--------------------
Indentation setup for shell type bash
```

GMM Training (2)

Terminal

```
#!/bin/bash
# Copyright 2012  Johns Hopkins University (Author: Daniel Povey)
# Apache 2.0

# Computes training alignments using a model with delta or
# LDA+MLLT features.

# If you supply the "--use-graphs true" option, it will use the training
# graphs from the source directory (where the model is).  In this
# case the number of jobs must match with the source directory.


# Begin configuration section.
nj=4
cmd=run.pl
use_graphs=false
# Begin configuration.
scale_opts="--transition-scale=1.0 --acoustic-scale=0.1 --self-loop-scale=0.1"
-uu-:---F1  align_si.sh    Top L1     (Shell-script[bash])----------------------
```

GMM Training (2)

Terminal

```
#!/bin/bash

# Copyright 2012  Johns Hopkins University (Author: Daniel Povey)
# Apache 2.0

# Begin configuration.
stage=-4 #  This allows restarting after partway, when something when wrong.
config=
cmd=run.pl
scale_opts="--transition-scale=1.0 --acoustic-scale=0.1 --self-loop-scale=0.1"
realign_iters="10 20 30";
num_iters=35    # Number of iterations of training
max_iter_inc=25 # Last iter to increase #Gauss on.
beam=10
careful=false
retry_beam=40
boost_silence=1.0 # Factor by which to boost silence likelihoods in alignment
power=0.25 # Exponent for number of gaussians according to occurrence counts
-uu-:---F1  train_deltas.sh   Top L1     (Shell-script[bash])--------------------
```

cluster-phones, compile-questions, build-tree

Terminal

```
if [ $stage -le -2 ]; then
  echo "$0: getting questions for tree-building, via clustering"
  # preparing questions, roots file...
  cluster-phones $context_opts $dir/treeacc $lang/phones/sets.int \
    $dir/questions.int 2> $dir/log/questions.log || exit 1;
  cat $lang/phones/extra_questions.int >> $dir/questions.int
  compile-questions $context_opts $lang/topo $dir/questions.int \
    $dir/questions.qst 2>$dir/log/compile_questions.log || exit 1;

  echo "$0: building the tree"
  $cmd $dir/log/build_tree.log \
    build-tree $context_opts --verbose=1 --max-leaves=$numleaves \
    --cluster-thresh=$cluster_thresh $dir/treeacc $lang/phones/roots.int \
    $dir/questions.qst $lang/topo $dir/tree || exit 1;

  $cmd $dir/log/init_model.log \
    gmm-init-model  --write-occs=$dir/1.occs  \
      $dir/tree $dir/treeacc $lang/topo $dir/1.mdl || exit 1;
-uu-:---F1  train_deltas.sh   48% L94    (Shell-script[bash])--------------------
```

GMM Training (4)

Terminal

```
  ### Triphone + LDA and MLLT
  # Training
  echo "Starting LDA+MLLT training."
  steps/align_si.sh  --nj $nj --cmd "$train_cmd"  \
      data/train data/lang exp/tri2a exp/tri2a_ali

  steps/train_lda_mllt.sh  --cmd "$train_cmd"  \
    --splice-opts "--left-context=3 --right-context=3" \
-uu-:---F1  run.sh          54% L116   (Shell-script[bash])----------------------
  ### Triphone + LDA and MLLT + SAT and FMLLR
  # Training
  echo "Starting SAT+FMLLR training."
  steps/align_si.sh  --nj $nj --cmd "$train_cmd" \
      --use-graphs true data/train data/lang exp/tri2b exp/tri2b_ali
  steps/train_sat.sh --cmd "$train_cmd" 4200 40000 \
      data/train data/lang exp/tri2b_ali exp/tri3b
  echo "SAT+FMLLR training done."

-uu-:---F1  run.sh          67% L138   (Shell-script[bash])----------------------
```

GMM Training (5)

Terminal

```
fi


if [ $stage -le 8 ]; then
  echo "Starting SGMM training."
  steps/align_fmllr.sh --nj $nj --cmd "$train_cmd" \
      data/train data/lang exp/tri3b exp/tri3b_ali

  steps/train_ubm.sh  --cmd "$train_cmd"  \
      600 data/train data/lang exp/tri3b_ali exp/ubm5b2

  steps/train_sgmm2.sh  --cmd "$train_cmd"  \
       5200 12000 data/train data/lang exp/tri3b_ali exp/ubm5b2/final.ubm exp/s\
gmm2_5b2
  echo "SGMM training done."

  (
  echo "Decoding the dev set using SGMM models"
-uu-:---F1   run.sh          80% L167    (Shell-script[bash])----------------------
```

Building an STT System with Kaldi

- Data preparation
  - Acoustic model training data
  - Pronunciation lexicon
  - Language model training data
- Basic GMM system building
  - Acoustic model training
  - Language model training
- Basic Decoding
  - **Creating a static decoding graph**
  - Lattice rescoring
- Basic DNN system building
- Going beyond the basics

Building HCLG (1)

Terminal

```
#!/bin/bash
# Copyright 2010-2012 Microsoft Corporation
#           2012-2013 Johns Hopkins University (Author: Daniel Povey)
# Apache 2.0

# This script creates a fully expanded decoding graph (HCLG) that represents
# all the language-model, pronunciation dictionary (lexicon), context-dependenc\
y,
-uu-:---F1  mkgraph.sh      Top L1     (Shell-script[bash])----------------------
if [[ ! -s $lang/tmp/LG.fst || $lang/tmp/LG.fst -ot $lang/G.fst || \
      $lang/tmp/LG.fst -ot $lang/L_disambig.fst ]]; then
  fsttablecompose $lang/L_disambig.fst $lang/G.fst | fstdeterminizestar --use-l\
og=true | \
    fstminimizeencoded | fstpushspecial | \
    fstarcsort --sort_type=ilabel > $lang/tmp/LG.fst || exit 1;
  fstisstochastic $lang/tmp/LG.fst || echo "[info]: LG not stochastic."
fi

-uu-:---F1  mkgraph.sh      46% L74    (Shell-script[bash])----------------------
```

Building HCLG (2)

Terminal

```
clg=$lang/tmp/CLG_${N}_${P}.fst

if [[ ! -s $clg || $clg -ot $lang/tmp/LG.fst ]]; then
  fstcomposecontext --context-size=$N --central-position=$P \
   --read-disambig-syms=$lang/phones/disambig.int \
   --write-disambig-syms=$lang/tmp/disambig_ilabels_${N}_${P}.int \
    $lang/tmp/ilabels_${N}_${P} < $lang/tmp/LG.fst |\
    fstarcsort --sort_type=ilabel > $clg
  fstisstochastic $clg  || echo "[info]: CLG not stochastic."
fi

if [[ ! -s $dir/Ha.fst || $dir/Ha.fst -ot $model  \
    || $dir/Ha.fst -ot $lang/tmp/ilabels_${N}_${P} ]]; then
  if $reverse; then
    make-h-transducer --reverse=true --push_weights=true \
      --disambig-syms-out=$dir/disambig_tid.int \
-uu-:---F1  mkgraph.sh      53% L87     (Shell-script[bash])----------------------
```

Building HCLG (3)

Terminal

```
if [[ ! -s $dir/HCLGa.fst || $dir/HCLGa.fst -ot $dir/Ha.fst || \
      $dir/HCLGa.fst -ot $clg ]]; then
  if $remove_oov; then
    [ ! -f $lang/oov.int ] && \
      echo "$0: --remove-oov option: no file $lang/oov.int" && exit 1;
    clg="fstrmsymbols --remove-arcs=true --apply-to-output=true $lang/oov.int $\
clg|"
  fi
  fsttablecompose $dir/Ha.fst "$clg" | fstdeterminizestar --use-log=true \
    | fstrmsymbols $dir/disambig_tid.int | fstrmepslocal | \
     fstminimizeencoded > $dir/HCLGa.fst || exit 1;
  fstisstochastic $dir/HCLGa.fst || echo "HCLGa is not stochastic"
fi

if [[ ! -s $dir/HCLG.fst || $dir/HCLG.fst -ot $dir/HCLGa.fst ]]; then
  add-self-loops --self-loop-scale=$loopscale --reorder=true \
    $model < $dir/HCLGa.fst > $dir/HCLG.fst || exit 1;
-uu-:---F1  mkgraph.sh      70% L112    (Shell-script[bash])----------------------
```

Building HCLG (4)

Terminal

```
fi

if [[ ! -s $dir/HCLG.fst || $dir/HCLG.fst -ot $dir/HCLGa.fst ]]; then
  add-self-loops --self-loop-scale=$loopscale --reorder=true \
    $model < $dir/HCLGa.fst > $dir/HCLG.fst || exit 1;

  if [ $tscale == 1.0 -a $loopscale == 1.0 ]; then
    # No point doing this test if transition-scale not 1, as it is bound to fai\
l.
    fstisstochastic $dir/HCLG.fst || echo "[info]: final HCLG is not stochastic\
."
  fi
fi

# keep a copy of the lexicon and a list of silence phones with HCLG...
# this means we can decode without reference to the $lang directory.


-uu-:---F1  mkgraph.sh      80% L125    (Shell-script[bash])----------------------
```

Decoding and Lattice Rescoring

Terminal

```
  echo "Starting SGMM training."
  steps/align_fmllr.sh --nj $nj --cmd "$train_cmd" \
      data/train data/lang exp/tri3b exp/tri3b_ali

  steps/train_ubm.sh  --cmd "$train_cmd"  \
      600 data/train data/lang exp/tri3b_ali exp/ubm5b2

  steps/train_sgmm2.sh  --cmd "$train_cmd"  \
       5200 12000 data/train data/lang exp/tri3b_ali exp/ubm5b2/final.ubm exp/s\
gmm2_5b2
  echo "SGMM training done."

  (
  echo "Decoding the dev set using SGMM models"
  # Graph compilation
  utils/mkgraph.sh data/lang_test exp/sgmm2_5b2 exp/sgmm2_5b2/graph
  utils/mkgraph.sh data/lang_big/ exp/sgmm2_5b2 exp/sgmm2_5b2/graph_big

  steps/decode_sgmm2.sh --nj $dev_nj --cmd "$decode_cmd" \
      --transform-dir exp/tri3b/decode_dev \
      exp/sgmm2_5b2/graph data/dev exp/sgmm2_5b2/decode_dev

  steps/lmrescore_const_arpa.sh  --cmd "$decode_cmd" \
      data/lang_test/ data/lang_big/ data/dev \
      exp/sgmm2_5b2/decode_dev exp/sgmm2_5b2/decode_dev.rescored

  steps/decode_sgmm2.sh --nj $dev_nj --cmd "$decode_cmd" \
      --transform-dir exp/tri3b/decode_dev \
      exp/sgmm2_5b2/graph_big data/dev exp/sgmm2_5b2/decode_dev.big
  echo "SGMM decoding done."
  ) &
fi

wait;
#score
for x in exp/*/decode*; do [ -d $x ] && grep WER $x/wer_* | utils/best_wer.sh; \
done
-uu-:---F1  run.sh          Bot L179    (Shell-script[bash])----------------------
```

Decoding and Lattice Rescoring

Terminal

```
  # Graph compilation
  utils/mkgraph.sh data/lang_test exp/sgmm2_5b2 exp/sgmm2_5b2/graph
  utils/mkgraph.sh data/lang_big/ exp/sgmm2_5b2 exp/sgmm2_5b2/graph_big

  steps/decode_sgmm2.sh --nj $dev_nj --cmd "$decode_cmd" \
      --transform-dir exp/tri3b/decode_dev \
      exp/sgmm2_5b2/graph data/dev exp/sgmm2_5b2/decode_dev

  steps/lmrescore_const_arpa.sh  --cmd "$decode_cmd" \
      data/lang_test/ data/lang_big/ data/dev \
      exp/sgmm2_5b2/decode_dev exp/sgmm2_5b2/decode_dev.rescored

  steps/decode_sgmm2.sh --nj $dev_nj --cmd "$decode_cmd" \
      --transform-dir exp/tri3b/decode_dev \
      exp/sgmm2_5b2/graph_big data/dev exp/sgmm2_5b2/decode_dev.big
  echo "SGMM decoding done."
  ) &
fi
-uu-:---F1   run.sh          88% L184    (Shell-script[bash])----------------------
```

steps/decode_sgmm2.sh

Terminal

```
#!/bin/bash

# Copyright 2012  Johns Hopkins University (Author: Daniel Povey).  Apache 2.0.

# This script does decoding with an SGMM system, with speaker vectors.
# If the SGMM system was
# built on top of fMLLR transforms from a conventional system, you should
# provide the --transform-dir option.

# Begin configuration section.
stage=1
transform_dir=    # dir to find fMLLR transforms.
nj=4 # number of decoding jobs.
acwt=0.1  # Just a default value, used for adaptation and beam-pruning..
cmd=run.pl
beam=13.0
gselect=15  # Number of Gaussian-selection indices for SGMMs.  [Note:
            # the first_pass_gselect variable is used for the 1st pass of
-uu-:---F1  decode_sgmm2.sh    Top L1     (Shell-script[bash])------------------
```

Building an STT System with Kaldi

- Data preparation
  - Acoustic model training data
  - Pronunciation lexicon
  - Language model training data
- Basic GMM system building
  - Acoustic model training
  - Language model training
- Basic Decoding
  - Creating a static decoding graph
  - **Lattice rescoring**
- Basic DNN system building
- Going beyond the basics

steps/lmrescore_const_arpa.sh

Terminal

```
if ! cmp -s $oldlang/words.txt $newlang/words.txt; then
  echo "$0: $oldlang/words.txt and $newlang/words.txt differ: make sure you kno\
w what you are doing.";
fi

oldlmcommand="fstproject --project_output=true $oldlm |"

-uu-:---F1  lmrescore_const_arpa.sh    55% L46     (Shell-script[bash])----------

if [ $stage -le 1 ]; then
  $cmd JOB=1:$nj $outdir/log/rescorelm.JOB.log \
    lattice-lmrescore --lm-scale=-1.0 \
    "ark:gunzip -c $indir/lat.JOB.gz|" "$oldlmcommand" ark:-  \| \
    lattice-lmrescore-const-arpa --lm-scale=1.0 \
    ark:- "$newlm" "ark,t:|gzip -c>$outdir/lat.JOB.gz" || exit 1;
fi

-uu-:---F1  lmrescore_const_arpa.sh    70% L57     (Shell-script[bash])----------
```

Building an STT System with Kaldi

- Data preparation
  - Acoustic model training data
  - Pronunciation lexicon
  - Language model training data
- Basic GMM system building
  - Acoustic model training
  - Language model training
- Basic Decoding
  - Creating a static decoding graph
  - Lattice rescoring
- **Basic DNN system building**
- Going beyond the basics

local/nnet3/run_ivector_common.sh

Terminal

```
if [ $stage -le 1 ]; then
    for datadir in train; do
      utils/perturb_data_dir_speed.sh 0.9 data/${datadir} data/temp1
      utils/perturb_data_dir_speed.sh 1.1 data/${datadir} data/temp2
      utils/combine_data.sh data/${datadir}_tmp data/temp1 data/temp2
      utils/validate_data_dir.sh --no-feats data/${datadir}_tmp
      rm -r data/temp1 data/temp2

      mfccdir=mfcc_perturbed
      steps/make_mfcc.sh --cmd "$train_cmd" --nj 17 \
        data/${datadir}_tmp exp/make_mfcc/${datadir}_tmp $mfccdir || exit 1;
      steps/compute_cmvn_stats.sh data/${datadir}_tmp exp/make_mfcc/${datadir}_\
tmp $mfccdir || exit 1;
      utils/fix_data_dir.sh data/${datadir}_tmp

      utils/copy_data_dir.sh --spk-prefix sp1.0- --utt-prefix sp1.0- data/${dat\
adir} data/temp0
      utils/combine_data.sh data/${datadir}_sp data/${datadir}_tmp data/temp0
-uu-:---F1  run_ivector_common.sh    5% L24    (Shell-script[bash])------------
```

local/nnet3/run_ivector_common.sh

Terminal

```
if [ $stage -le 5 ]; then
  steps/online/nnet2/train_diag_ubm.sh --cmd "$train_cmd" --nj 16  --num-frames\
 200000 \
     data/train_hires 256 exp/nnet3/tri5b exp/nnet3/diag_ubm || exit 1
fi

if [ $stage -le 6 ]; then
  # even though $nj is just 10, each job uses multiple processes and threads.
steps/online/nnet2/train_ivector_extractor.sh --cmd "$train_cmd" \
    --nj 10 --num-processes 1 --num-threads 2  --ivector-dim 50\
    data/train_hires exp/nnet3/diag_ubm exp/nnet3/extractor || exit 1;
fi

if [ $stage -le 7 ]; then
  # having a larger number of speakers is helpful for generalization, and to
  # handle per-utterance decoding well (iVector starts at zero).
  steps/online/nnet2/copy_data_dir.sh --utts-per-spk-max 2 data/train_hires \
-uu-:---F1  run_ivector_common.sh   69% L75    (Shell-script[bash])------------
```

local/nnet3/run_ivector_common.sh

Terminal

```
fi

if [ $stage -le 7 ]; then
  # having a larger number of speakers is helpful for generalization, and to
  # handle per-utterance decoding well (iVector starts at zero).
  steps/online/nnet2/copy_data_dir.sh --utts-per-spk-max 2 data/train_hires \
    data/train_hires_max2 || exit 1

  steps/online/nnet2/extract_ivectors_online.sh --cmd "$train_cmd" --nj 16\
    data/train_hires_max2 exp/nnet3/extractor exp/nnet3/ivectors_train || exit \
1
fi

if [ $stage -le 8 ]; then
    steps/online/nnet2/extract_ivectors_online.sh --cmd "$train_cmd" --nj 6 \
      data/dev_hires exp/nnet3/extractor exp/nnet3/ivectors_dev || exit 1
fi
```

-uu-:---F1 **run_ivector_common.sh** 83% L87 (Shell-script[bash])----------

steps/nnet3/tdnn/make_configs.py

Terminal

```
local/nnet3/run_ivector_common.sh --stage $stage || exit 1;

if [ $stage -le 9 ]; then
  echo "$0: creating neural net configs";

  # create the config files for nnet initialization
  python steps/nnet3/tdnn/make_configs.py  \
    --feat-dir data/train_hires \
    --ivector-dir exp/nnet3/ivectors_train \
    --ali-dir exp/nnet3/tri3b_ali_sp \
    --relu-dim 256 \
    --splice-indexes=" -2,-1,0,1,2  -1,0,1  -1,0,1  -1,0,1  -1,0,1 -1,0,1 0 " \
    --use-presoftmax-prior-scale true \
   $dir/configs || exit 1;
fi
```

-uu-:---F1 **run_tdnn.sh**   29% L40   (Shell-script[bash])----------

steps/nnet3/tdnn/make_configs.py

Terminal

```
  /Users/sanjeev/Desktop/Conference Travel/SLTU 2016/Tutorial/kaldi/egs/iban/s5\
/exp/nnet3/tdnn_1/configs:
  total used in directory 88 available 213368700
  drwxr-xr-x  13 sanjeev  staff   442 May  3 12:39 .
  drwxr-xr-x  26 sanjeev  staff   884 May  3 13:26 ..
  -rw-r--r--   1 sanjeev  staff   333 May  3 12:35 init.config
  -rw-r--r--   1 sanjeev  staff  1454 May  3 12:35 layer1.config
  -rw-r--r--   1 sanjeev  staff  1185 May  3 12:35 layer2.config
  -rw-r--r--   1 sanjeev  staff  1185 May  3 12:35 layer3.config
  -rw-r--r--   1 sanjeev  staff  1185 May  3 12:35 layer4.config
  -rw-r--r--   1 sanjeev  staff  1185 May  3 12:35 layer5.config
  -rw-r--r--   1 sanjeev  staff  1185 May  3 12:35 layer6.config
  -rw-r--r--   1 sanjeev  staff  1122 May  3 12:35 layer7.config
  lrwxr-xr-x   1 sanjeev  staff    10 May  3 12:39 lda.mat -> ../lda.mat
  lrwxr-xr-x   1 sanjeev  staff    29 May  3 12:39 presoftmax_prior_scale.vec -\
> ../presoftmax_prior_scale.vec
  -rw-r--r--   1 sanjeev  staff   140 May  3 12:35 vars

-uuu:%%-F1  configs            All L5     (Dired by name)----------------------
Loading dired...done
```

steps/nnet3/train_dnn.py

Terminal

```
steps/nnet3/train_dnn.py --stage $train_stage \
  --cmd="$decode_cmd" \
  --trainer.optimization.num-jobs-initial 2 \
  --trainer.optimization.num-jobs-final 4 \
  --trainer.num-epochs 4 \
  --trainer.add-layers-period 1 \
  --feat.online-ivector-dir exp/nnet3/ivectors_train\
  --feat.cmvn-opts "--norm-means=false --norm-vars=false" \
  --trainer.num-epochs 2 \
  --trainer.optimization.initial-effective-lrate 0.005 \
  --trainer.optimization.final-effective-lrate 0.0005 \
  --trainer.samples-per-iter 120000 \
  --cleanup.preserve-model-interval 10 \
  --feat-dir data/train_hires \
  --ali-dir exp/nnet3/tri3b_ali_sp \
  --lang data/lang \
  --dir=$dir  || exit 1;
-uu-:---F1  run_tdnn.sh     51% L60     (Shell-script[bash])----------------------
```