CRIM

CENTRE DE
RECHERCHE INFORMATIQUE
DE MONTRÉAL

# USING KALDI TO TRANSCRIBE ATC CONVERSATIONS

VISHWA GUPTA

JUNE 10, 2019

crim.ca

PRINCIPAL PARTENAIRE FINANCIER

Économie, Science et Innovation Québec

# AIR TRAFFIC CONTROL (ATC) TASK OUTLINE

## 1. TRANSCRIBE CONVERSATIONS BETWEEN CONTROLLERS AND PILOTS

- Example conversation:  clear takeoff three two right Easy two six one Quebec

## 2. DATA PROVIDED BY ATC ORGANIZERS:

- 28000 transcribed audio files representing speaker turns (40 hours of audio)
- How to use this data to create a speech transcription system?

# STEP 1: CREATE KALDI DATA STRUCTURE

## 1. CREATE 4 FILES:

### wav.scp

- 006dxvWC0xfc4I65 Wavfiles/006dxvWC0xfc4I65.wav
- 00AVfATRNNwTNzQ0 Wavfiles/00AVfATRNNwTNzQ0.wav

### text

- 006dxvWC0xfc4I65 Easy five seven three charlie contact Bordeaux control
- 00AVfATRNNwTNzQ0 KLM seven three november say speed

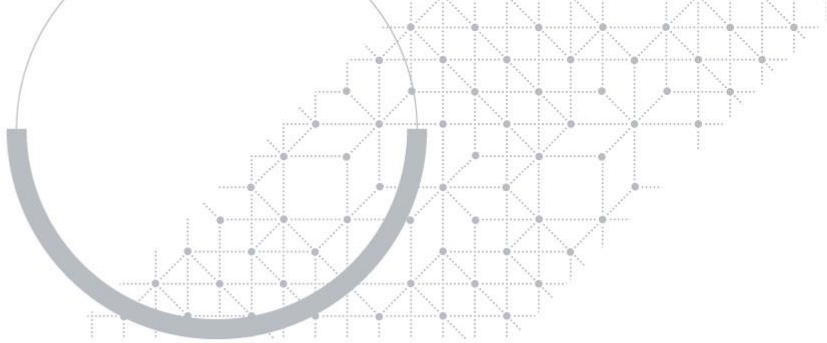
### utt2spk

006dxvWC0xfc4I65 006dxvWC0xfc4I65

00AVfATRNNwTNzQ0 00AVfATRNNwTNzQ0

### spk2utt

# SEGMENTS FILE



1. SOMETIMES THE AUDIO FILE IS DIVIDED INTO MANY SEGMENTS TO REMOVE MUSIC NOISE ETC.

UTT_ID1 WAV_ID1 0.21 5.67

UTT_ID2 WAV_ID1 6.67 26.23

---

# DICTIONARY

1. dictionary is stored in data/local/dict/lexicon.txt

**alpha AE L F AH**

**b IY**

**QHN K Y UW EY CH EH N**

**QHN K Y UW EY CH N**

# lang DIRECTORY

1. Lang directory is created by the script utils/prepare_lang.sh using data/local/dict

-- This script adds word-position-dependent phones and constructs a host of other derived files

-- dict directory has following files: extra_questions.txt lexicon.txt nonsilence_phones.txt optional_silence.txt silence_phones.txt

-- nonsilence_phones.txt has list of all the non silence phones in the dictionary:

AA

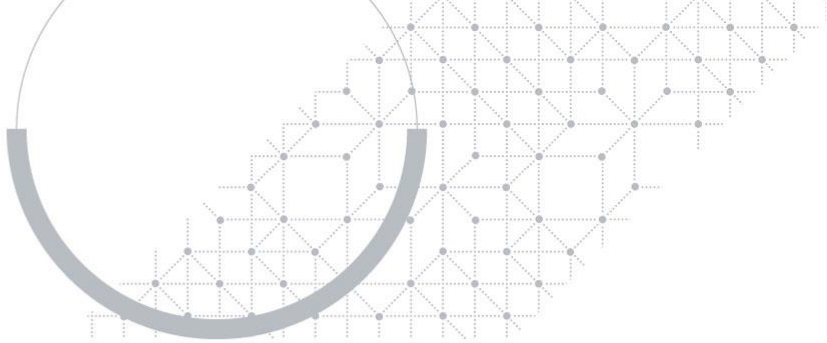
AE

AH

AO

---

# Language Models



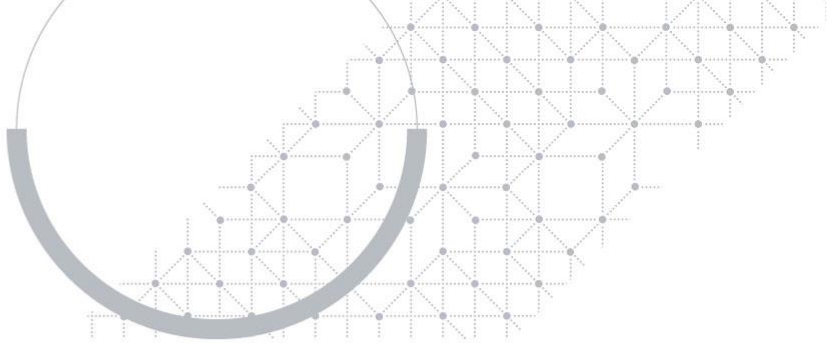
1. Language models are created using the SRILM toolkit. It has an executable ngram that is used to generate n-gram language models in ARPA format. Usually a 3-gram (or trigram) language model is used for search and a 4-gram for rescoring. The input to ngram is a text file containing sentences from the language.
2. The script local/train_lms_srilm.sh builds an SRILM language model in ARPA format. It needs training and validation data in kaldi text format.
3. Script utils/format_lm.sh converts the ARPA-format language models to FSTs. These are stored in directories data/lang_tgpr or data/lang_qg
4. Once the dict, lang and lang_tgpr directoris have been generated, we can create HMM/GMM models and run decoding.

# GENERATE GMM/HMM MODELS

1. A script run.sh in s5 directory generates the GMM/HMM models

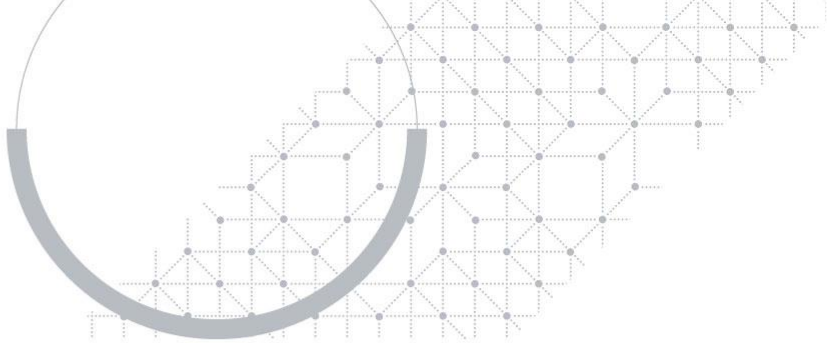
   - steps/make_mfcc.sh --  generates mfcc features
   - Steps/compute_cmvn_stats.sh  -- computes cepstral mean and variance per speaker for feature normalization
   - Utils/subset_data_dir_tr_cv.sh  -- divides the kaldi directory into two subdirectories: training and development sets
   - There are many steps involved in generating the GMM/HMM models.
   - From the trained GMM/HMM models, we generate alignments used for DNN training
   - Steps/align_fmllr script generates the alignments to be used for training DNN models

# TRAIN DNN MODELS



1. Initial DNNs were simple feedforward multi-layer neural nets
   - Each hidden layer is identical followed by sigmoid or rectified linear or tanh nonlinearity
2. Recursive neural nets giving best results now for both acoustic & language models
   - Bidirectional LSTM (BLSTM) models seem to give the best results
3. Factored TDNN (TDNN-F) are now giving the best results for real-time systems
   - Weight matrix AxB is factored into AxC and CxB where C is semi-orthogonal
   - C is of a smaller dimension and acts as a bottleneck layer and reduces the total parameters
   - The training script is in local/chain/run_tdnn.sh (in the kaldi download for say swbd)

# TRAINING SCRIPTS FOR DNN MODELS



1. 3 different nnet structures in kaldi: nnet1, nnet2 and nnet3
   - All current work is probably only in nnet3
   - I have not used nnet1 or nnet2 scripts in a long time
2. BLSTM script is local/nnet3/run_lstm.sh --lstm-delay " [-1,1] [-2,2] [-3,3] " --label-delay 0
   - Bidirectional LSTM (BLSTM) models seem to give the best results
3. Factored TDNN (TDNN-F) script is
   - swbd/s5c/local/chain/run_tdnn.sh

# I-VECTORS

1. I-VECTORS REPRESENT SPEAKER CHARACTERISTICS IN A FIXED DIMENSIONAL SPACE (USUALLY 100-DIMENSIONAL):
   - These i-vectors are added as an additional input to the DNN
   - Train i-vector extractor
   - Generate i-vectors for each speaker (or utterance) using the i-vector extractor
   - i-vector training scripts are in local/nnet3/run_ivector_common.sh

# SAMPLE TRAINING SCRIPT ANALYSIS

## 1. local/nnet3/run_lstm.sh

- local/nnet3/run_ivector_common.sh -- speed perturb data, compute MFCC, generate alignments, compute diagonal UBM, train ivector extractor, extract i-vectors
- steps/nnet3/lstm/make_configs.py -- generate neural net configs for LSTM models
- steps/nnet3/train_rnn.py -- train LSTM models
  - Generate egs (randomized training data split into batches)
  - Compute preconditioning matrix for features (lda.mat)
  - Train the acoustic models for n epochs
- steps/nnet3/decode.sh -- decode the dev set using the trained LSTM model.