Building a Speech Recognizer

IAP 2016: Technologies for Speech and Language Processing

Mandy Korpusik and Tuka Al Hanai
Friday 15th Jan 2016
Today

1. Acoustic Phonetics
2. Language Modeling
3. Acoustic Modeling
4. **Building a Recognizer**
5. Bringing Speech Technologies to the World
Overview

1. Forced Alignments
2. Decoding
   a. Finite State Transducer
   b. Viterbi Decoding
3. Activity: Analyzing Word Error Rate
4. Extra Activity: Calculating Word Error Rate
5. Extra Activity: Decoding with Kaldi
Speech Recognition Pipeline

Audio → Feature Extraction → Recognizer → Text

Lexical Model
Language Model
Acoustic Model
The sky that morning was clear and bright blue.

/θɛs kɑɪ thət mɔrˈnɪŋ wəz klɪər ənd bɹəɪt blu/
Alignments

We can see where sounds start/end, but we have experience!
Steps

1. Divide audio uniformly
   a. flat-start initialization
2. Model each phone
   independant of context
3. Generate new alignments
   with learned monophone models
4. Train context dependant models
5. Generate new alignments
   with triphone models
6. iterate.

Evaluation tells us models/alignments are better!
Kaldi recipe (.run.sh)

1. steps/make_mfcc.sh -- Feature Extraction
2. steps/train_mono.sh -- Training Monophone Models
3. steps/align_si.sh -- Forced Alignments
4. steps/train_deltas.sh -- Training Triphone Models
5. … iterate above steps -- Iterations of Triphone Models

Evaluation

6. utils/mkgraph.sh
7. steps/decode.sh
Triphone Model (Context Dependant Model)
Triphone Model (Context Dependant Model)
How do we tie states together?

Decision Tree

$N^3$ possible triphones, but not all can/will be observed.
Forced Alignments
Forced Alignments

This assumes I know the pronunciation of words.

But what if I don’t know …? →
What if I don’t have a dictionary?

<table>
<thead>
<tr>
<th>Phonetic Dictionary</th>
<th>Graphemic Dictionary</th>
<th>Still works decently well. Some languages better than others.</th>
</tr>
</thead>
<tbody>
<tr>
<td>blue : /b l e w/</td>
<td>blue : /b l u e/</td>
<td>e.g. Mandarin? Uses character with no clear mapping to phones. :-</td>
</tr>
</tbody>
</table>
Kaldi recipe (./run.sh)

1. `steps/make_mfcc.sh` -- Feature Extraction
2. `steps/train_mono.sh` -- Training Monophone Models
3. `steps/align_si.sh` -- Forced Alignments
4. `steps/train_deltas.sh` -- Training Triphone Models
5. … iterate above steps -- Iterations of Triphone Models

Evaluation

6. `utils/mkgraph.sh`
7. `steps/decode.sh`
Kaldi recipe (./run.sh)

1. `steps/make_mfcc.sh` -- Feature Extraction
2. `steps/train_mono.sh` -- Training Monophone Models
3. `steps/align_si.sh` -- Forced Alignments
4. `steps/train_deltas.sh` -- Training Triphone Models
5. ... iterate above steps -- Iterations of Triphone Models

Evaluation

6. `utils/mkgraph.sh`
7. `steps/decode.sh` → Let’s look at these last two step
Phonotactics

Phoneme sequence constraints

- /pt/ can't be in word beginning
- /tp/ not a word ending
- /tpt/ not a legitimate sequence

Legal in English, but not a word

- /stremp/
- /strunk/

LANGUAGE-SPECIFIC CONSTRAINTS

Languages differ in the kinds of onsets they allow:

<table>
<thead>
<tr>
<th></th>
<th>/kn/</th>
<th>/skw/</th>
<th>/sb/</th>
<th>/vr/</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>German</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>French</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Italian</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>
Phonotactics

Phoneme sequence constraints

- /pt/ can’t be in word beginning
- /tp/ not a word ending
- /tpt/ not a legitimate sequence

Legal in English, but not a word

- /stremp/
- /strunk/

Can you think of other non-English English words?
Finite State Transducer

Composing pronunciation and grammar into a search space.

Constrained over grammar.
Finite State Transducer

Composing pronunciation and grammar into a search space.

Constrained over dictionary.

Constrained over grammar.
Finite State Transducer

Composing pronunciation and grammar into a search space.

Triphone GMM-HMM

Constrained over dictionary.

Constrained over grammar.
Finite State Transducer

Generates a sequence of characters/words. Can also be weighted.

(words $\rightarrow$ phonemes)

![Diagram](image-url)
Finite State Transducer

Generates a sequence of characters/words. Can also be weighted.

\[(\text{words} \rightarrow \text{phonemes}) \quad (\text{phonemes} \rightarrow \text{phones})\]
Finite State Transducer

Generates a sequence of characters/words. Can also be weighted.

\[(\text{words} \to \text{phonemes}) \circ (\text{phonemes} \to \text{phones}) = (\text{words} \to \text{phones})\]
Finite State Transducer (.steps/mkgraph.sh)

Composed as $H \circ C \circ L \circ G$ (in KALDI)

- $H =$ HMM Definitions  (input: transition id, output: tri-phone)
- $C =$ Context dependency  (input: tri-phone, output: phone)
- $L =$ Lexicon  (Dictionary, input: phone, output: word)
- $G =$ Grammar  (Language Model, input = output: word)
Finite State Transducer
Example of a lexicon. (Determinized and minimised)
Viterbi Decoding (./steps/decode.sh)

Finding best path for given observations (features - MFCCs)
1. Each cell in step N maintains pointer to best source N-1
2. Traceback path from highest probability
The entries in the individual state columns for the Viterbi algorithm. Each cell keeps the probability of the best path so far and a pointer to the previous cell along that path. Backtracing from the successful last word (*the*), we can reconstruct the word sequence *I need the.*
Figure 7.12 The use of $N$-best decoding as part of a two-stage decoding model. Efficient but unsophisticated knowledge sources are used to return the $N$-best utterances. This significantly reduces the search space for the second pass models, which are thus free to be very sophisticated but slow.
Evaluating Performance

Word Error Rate (WER) Metric

\[
\text{WER} = \frac{(S + I + D)}{N}
\]

\[
S = \# \text{ of Substitutions}
\]

\[
I = \# \text{ of Insertions}
\]

\[
D = \# \text{ of Deletions}
\]

\[
N = \text{Total \# of words in reference}
\]

Word Accuracy = 1 - WER (but we rarely report this)
Activity: Analyze WER

Look at the files in this folder. Contains files that already had WER calculations run on Voxforge monophone models: http://web.mit.edu/org/s/speechiap/www/wer-eval.zip

- 10.ref (reference)
- 10.hyp (hypothesis)
- 10.hyp.pra (alignment between ref and hyp)
- 10.hyp.dtl (details on insertions, deletions, and substitutions)

What kind of errors appear? What is their nature?
What is the most common substitution?
Notice the length of the utterances being decoded.
Can you guess what is an OOV (Out-of-Vocabulary) word?
Any other observations?
Examples

REF:  SHE WAS EVEN MORE BEAUTIFUL THAN WHEN I SAW HER BEFORE
HYP:  SHE WAS EVEN MORE ********* UNEVEN ONE I SAW HER BEFORE
Eval:                   D         S      S

Substitutions - short/function words

1:  14  ->  A ==> THE
2:  10  ->  AND ==> CAN
3:  10  ->  AND ==> IN
4:   8  ->  A ==> TO
5:   8  ->  HE ==> SHE
6:   7  ->  THEIR ==> THE
7:   6  ->  IT ==> TO
8:   6  ->  THE ==> A
9:   6  ->  THE ==> AND
10:  6  ->  THERE ==> THE
WER Observations

Shortcomings:

- Does not inform us about the nature of errors made.
  - A lot of substitutions of function words (and, of, the)
  - Short rare words missed, acoustic modeling misses them?
- Might want to know if it’s due to
  - poor language modeling
  - bad pronunciations
  - accented speech
  - poor acoustic modeling
  - reference transcription error
## Results

<table>
<thead>
<tr>
<th>Features</th>
<th>Model</th>
<th>WER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFCCs</td>
<td>Monophone Model</td>
<td>17.0</td>
</tr>
<tr>
<td>MFCCs</td>
<td>Triphone Model (1st iter)</td>
<td>6.9</td>
</tr>
<tr>
<td>MFCCs + delta + delta-delta</td>
<td>Triphone Model (2nd iter)</td>
<td>6.5</td>
</tr>
<tr>
<td>MFCCs + delta + delta-delta + LDA + MLLT</td>
<td>Triphone Model (3rd iter)</td>
<td>6.5</td>
</tr>
</tbody>
</table>

**Significant Improvement!**
## Results in Literature

<table>
<thead>
<tr>
<th>IBM Watson</th>
<th>Google</th>
<th>MIT CSAIL SLS</th>
<th>Human</th>
</tr>
</thead>
<tbody>
<tr>
<td>8%</td>
<td>15 - 50 %</td>
<td>60s</td>
<td>4%</td>
</tr>
<tr>
<td>Deep Neural Networks</td>
<td>GMM-HMM</td>
<td>Deep Neural Networks</td>
<td>?</td>
</tr>
<tr>
<td>NIST Switchboard (Telephone Conversation)</td>
<td>Google Voice Search, Youtube videos (1400 hours)</td>
<td>IARPA - Babel, Low Resource Languages (10 - 80 hours)</td>
<td>General Scenarios</td>
</tr>
</tbody>
</table>
Is High WER Useless?

We can still do topic modeling.

Understand the theme and nature of the conversation.
Putting it all together
Extra Activity: Decode Audio

1. Download: (still uploading...)
   a. You will need 1.5GB of space.

2. Save it in your $KALDI_ROOT dir:

3. Execute ./run_decode.sh
   a. njobs=1 (change according to # of CPU cores)

4. The search graph has already been composed: $KALDI_ROOT/exp/*/graph

5. You will be decoding the following models
   a. Monophone Models
   b. Triphone Models (3 of them)

6. Look at the WER in this file: $KALDI_ROOT/exp/*/decode/wer_*
Extra Activity: Calculate WER


Will compare reference and hypothesis files to calculate:

- Subs, ins, dels, and WER
- Prints all subs, ins, dels (.dtl)
- Prints alignment between utterances (.pra)
What is Kaldi?

Speech Recognition Toolkit popularly used by researchers.

Can build:

- Dictionaries
- Language Models
- Acoustic Models
  - GMM, Deep Neural Networks
- Build a recognizer
- Setup an online decoder

Has ‘recipes’ for many many corpora and setups.

Single platform for publishing and evaluating system designs.

Written in C++, executed via bash scripts.

http://kaldi-asr.org/index.html
Next Class

Bringing Speech Technologies to the World
Steps

1. Flat-start monophone set created
   a. divide speech into phone segments of equal length
   b. single Gaussian model each phone base
2. Build models
3. Do force alignment between audio and transcription
4. Build new models using this new alignment
5. Iterate
Amazon

Alexa
Build engaging voice experiences for your services and devices

Alexa Skills Kit (ASK) Developer Preview
A free SDK that lets you easily add new voice capabilities

Alexa Voice Service (AVS) Developer Preview
Bring voice capabilities to your connected device

The Alexa Fund
$100 million in investment to fuel voice technology innovation
IBM Watson brings together a set of transformational technologies to drive optimized outcomes.

1. Understands natural language and human speech.
2. Generates and evaluates hypotheses for better outcomes.
3. Adapts and learns from user selections and responses.

Combining Knowledge & Data-Driven Technologies for a Learning Healthcare System.

- Data
  - Patients
  - Providers
  - Payment

Knowledge- and Data-Driven Personalized Evidence Delivery

Personalized Evidence & Insights

Data-Driven Analytics for Personalized Evidence Generation
IBM Developer Cloud

Nuance

Dragon NaturallySpeaking for developers

Learn more about how to add speech recognition to in-house and commercial applications or workflows

Learn More about Dragon for Developers

Top Imaging OCR Toolkit for Windows, Linux and Mac

OmniPage SDK has everything you need to add powerful imaging, recognition, conversion and PDF features to your applications.

More About OmniPage Capture SDK

Nuance Mobile Solutions for Developers

Speech-enable your mobile app within minutes with the Dragon Mobile SDK.

More About Nuance Mobile Developer SDK

Nuance Healthcare Solutions for Developers

The 360 | Development Platform allows you to embed medical voice and understanding functionality in your applications.

360 | Development Platform

PowerShare Innovation
Useful Links

- Build an iOS app like Siri

- Google Speech API
  - [https://www.npmjs.com/package/google-speech-api](https://www.npmjs.com/package/google-speech-api)

- HTML5 Speech API

- IBM Watson Speech API