Speech to Text Translation Using Google Speech Commands

Hyung Lee and Amita C. Patil
hyunglee@stanford.edu, amita2@stanford.edu

CS 229 (Fall 2019)
Final Project

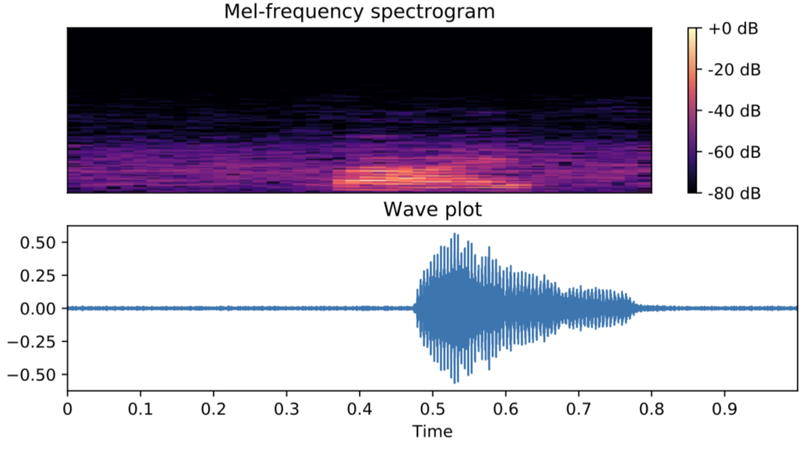
Objective

- Audio Commands to Text Translation on the Edge
- We use three ML techniques: Multinomial Logistic Regression, Hidden Markov Models with Mixture of Gaussians, CNN and compare with oracle

Dataset and Features

- Google speech commands[1]: 105,829 audio files of people speaking single-word commands
- 20 core words e.g. “zero”, “one”, “yes”, “no”, “stop”, “go” which most speakers say 5 times
- 10 auxillary words e.g. “bird”, ”sheila” which most speakers say once

MFCC[2] and waveplot of a person saying “FIVE”



Models

N states: $S = \{S_1, S_2, \dots, S_N\}$ M observation symbols per state: $v = \{v_1, v_2, \dots, v_M\}$

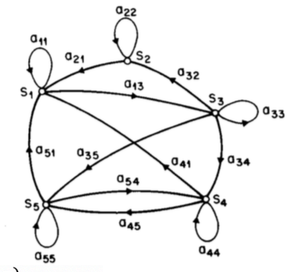
State transition probability:

$A_{ij} = \{a_{ij}\}$ where $a_{ij} = P[q_{t+1} = S_j | q_t = S_i]$; $1 \le i$ and $1 \le j \le N$

Observation symbol probability in state j:

$B = \{b_j(k) = P[v_k \; at \; t \mid q_t = S_j]$, $1 \le j \le N$ and $1 \le k \le M$

Initial state distribution: $\pi = \{\pi_i\}$ where $\pi_i = P[q_1 = S_i]$, $1 \le i \le N$



HMM with 5 states [3]

- Given model parameters *N*, *A*, *B*, and π
  HMM can generate observation sequence $O = O_1 O_2 \dots O_T$
- Given observation sequence $O = O_1 O_2 \dots O_T$ and model $\lambda = (A, B, \pi)$
  We can compute $P[O|\lambda]$ by summing joint probability over all possible state sequences q

$$P[O|\lambda] = \sum_{q_1, q_2, \dots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \dots a_{q_{T-1} q_2} b_{q_T}(Q_T)$$

- Training involves adjusting model parameters $\lambda = (A, B, \pi)$ to maximize $P[O|\lambda]$

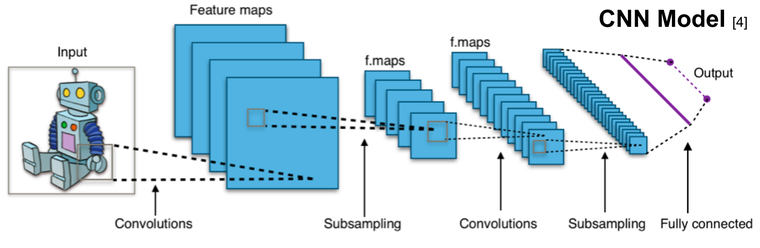
$$\overline{\pi_i} = expected \; \# \; of \; times \; in \; S_i \; at \; t = 1$$

$$\overline{a_{ij}} = \frac{expected \; \# \; of \; transitions \; from \; state \; S_i \; to \; S_j}{expected \; \# \; of \; transitions \; from \; state \; S_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

where $\xi_t(i,j) = P[q_t = S_i | q_{t+1} = S_j | O, \lambda]$, $\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$

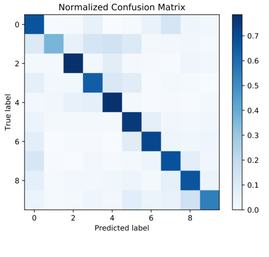
$$\overline{b_j}(k) = \frac{expected \; \# \; of \; times \; in \; state \; j \; and \; observing \; symbol \; v_k}{expected \; \# \; of \; times \; in \; state \; S_j} = \frac{\sum_{t=1 \; s.t. \; O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

CNN Model [4]

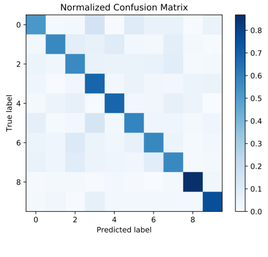


Input is a tensor with shape (number of examples) x (width) x (height) x (depth)
Convolutional layers turn the input into a feature map using kernels and receptive fields
Pooling employed as a means of downsampling
Fully connected layer and loss layer produce classes

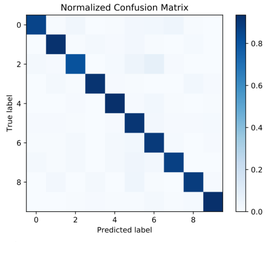
HMMGMM: 10 #’s



HMMGMM: 10 command words



CNN: 10 #’s



Results On Test Set

Model	Dataset	Accuracy	Precision (weighted)	Recall (weighted)	F1-score (weighted)	# of test examples
Multinomial LR	GSC v2 numbers 0 to 9	0.23	0.39	0.17	0.23	7782
HMM-GMM	GSC v2 numbers 0 to 9	0.66	0.67	0.66	0.65	4107
HMM-GMM	GSC v2 command words	0.65	0.65	0.65	0.65	2791
HMM-GMM	Our voice numbers 0 to 9	0.83	0.80	0.83	0.79	30
CNN	GSC v2 numbers 0 to 9	0.9	0.9	0.9	0.9	7782

As of late 2017, Google boasted a 95%word accuracy rate for U.S. English; the highest out of all the voice-assistants currently out there. This translates to a 4.9%word error rate – making Google the first to fall below the 5% threshold.

Summary / Future Work

- Multinomial Logistic Regression is too simplistic, does not identify speech commands based on MFCC features
- HMM with Mixture of Gaussians has better accuracy uses relatively small number of model parameters and works better on small dataset (one speaker, 130 examples), thus more suitable for edge devices
- Our CNN model was able to achieve accuracy of 90% on 10 command words
- Given more time we would try to improve accuracy on ALL google speech commands
  - By combining probabilistic HMM model with NN, and
  - Using RNN/LSTM

References:

1. (Aug. 2017). Google ai blog: Launching the speech commands dataset, [Online] https://ai.googleblog.com/2017/08/launching-speech-commands-dataset.html
2. Muda Lindasalwa et a;.,“ Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques”, Journal Of Computing, Volume 2, Issue 3, pp 138-143, ISSN 2151-9617, March 2010.
3. L. Rabiner and B. Juang, “An introduction to hidden markov models,” IEEE ASSP Magazine, vol. 3, no. 1, pp. 4–16, Jan. 1986.
4. O. A.-H. et al, “Convolutional neural networks for speech recognition,”IEEE/ACM TRANSACTIONS ON AUDIO,SPEECH, AND LANGUAGE PROCESSING, vol. 22, 2014. [Online] https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/CNN_ASLPTrans2-14.pdf
5. https://en.wikipedia.org/wiki/Convolutional_neural_network#Design
6. https://medium.com/manash-en-blog/building-a-dead-simple-word-recognition-engine-using-convnet-in-keras-25e72c19c12b

Link to you tube video

https://youtu.be/vq1I5nr3zWs