DSP 2019 Spring
HW2-1
Recognition of Spoken Chinese Digits with HTK

- Training Process:
  - For exact training procedure, MFCC extraction configuration, and HMM modeling, please refer to “03_training.sh”, “lib/config.cfg”, “lib/HCopy.cfg”, the output “hmm/models”, and other configuration files for more details.
    The major changes are:
    - More states in HMMs.
    - Split MFCC into streams when feeding them to HMM.
    - Refined increasement of Gaussian mixtures.
    - Untie some unnecessary tie and human-made adjustments s.t. the model is more data-driven instead of human-supervised.

```
MU +1 {}
 Mixup by 1 components per stream
  Mean GC = 25.621929, Std Dev GC = 26.080355
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
          jiou : mixup 3 -> 4
 MU: Number of mixes increased from 36 to 48

MU +1 {}
 Mixup by 1 components per stream
  Mean GC = 25.692240, Std Dev GC = 26.018152
           sil : mixup 3 -> 4
           sil : mixup 3 -> 4
           sil : mixup 3 -> 4
           sil : mixup 3 -> 4
           sil : mixup 3 -> 4
           sil : mixup 3 -> 4
           sil : mixup 3 -> 4
           sil : mixup 3 -> 4
           sil : mixup 3 -> 4
 MU: Number of mixes increased from 27 to 36

MU +1 {}
 Mixup by 1 components per stream
  Mean GC = 25.789660, Std Dev GC = 26.002577
            sp : mixup 2 -> 3
            sp : mixup 2 -> 3
            sp : mixup 2 -> 3
 MU: Number of mixes increased from 6 to 9
```

Figure 1: last few "MU" in command "HHEd"; e.g. there are 3 streams 4 effective states in HMM “jiou”, thus here we have 12 lines of “mixup”.

  - I use the default vector format, which is “MFCC_Z_E_D_A”, since this exact MFCC 39-d format is said to have good performance during lectures, and adding more information or like 3rd difference to make it 52-d MFCC or choosing not to use MFCC would need lots of heavy modification in every configuration file.

- As indicated in HTK handbook ch.7, we could set the HMM to accept "streams", such that
  $$b_j(o_t) = \prod_{s \in streams} (b_{j_s}(o_{t_s}))$$
  (check ch.7.1 eqn. (7.1) for more details)
  Since the 39-d MFCC comes from {the data itself}, {the first difference of the data}, and {the second difference of the data}, I think it makes sense to split the 39-d MFCC into 3 streams where each stream is 13-d width: the streams represent "now", "first derivative", and "second derivative" respectively.

- The default header config file ties one state of HMM "sp" with one state of HMM "sil"; untying it, together with some other parameter change, yields better results; maybe it's because that the short silence in between utterances is different from the silence of when we finish a phrase: there may be tiny noise produced by human speech system during the former silence, since we need to change the shape of mouth, the position of tongue or teeth, etc, to make the next utterance, while during the latter silence we don't need such move and it's hence possible that the tiny noises of motion of our speech system is consequentially smaller.

- There are some human-made adjustments in "lib/sil1.hed"; I removed most of them s.t. the model is more data-driven: only the line
  "MU 2 {sil.state[2-4].mix}"
  is left.

- Since my models have 3 streams instead of the default 1, the corresponding headers "lib/mix2_10.hed" and "lib/mixAdd2_10.hed" have to be adjusted correspondingly such that the increase of Gaussian mixtures would take effect on all 3 streams instead of just 1; additionally, the latter now shall be called "lib/mixAdd1_10.hed" since I changed it to add just 1 mixture instead of 2.

- It's weird seeing that "bin/models_1mixsil.c" claims to produce a "5 states, 10 mixtures" HMM in its comments whereas what it actually do is just producing a "5 states, 1 mixtures" HMM: by default all it does is putting the "sil" HMM into "hmm/models" with extra "<NUMMIXES> 1" tag while the remaining part is copied from the default "hmm/hmmdef" file.
  I mean, by the handbook, the default mixture count is already 1. I don't see related treatments in "03_training.sh" as well, however it make it 3 mixtures with the header "mix2_10.hed".
  Therefore I changed the "sil" part in "mix2_10.hed" to be
  "MU 2 {sil.state[2-4].stream[1-3].mix}",
  making the adding-mixture operation more graceful.

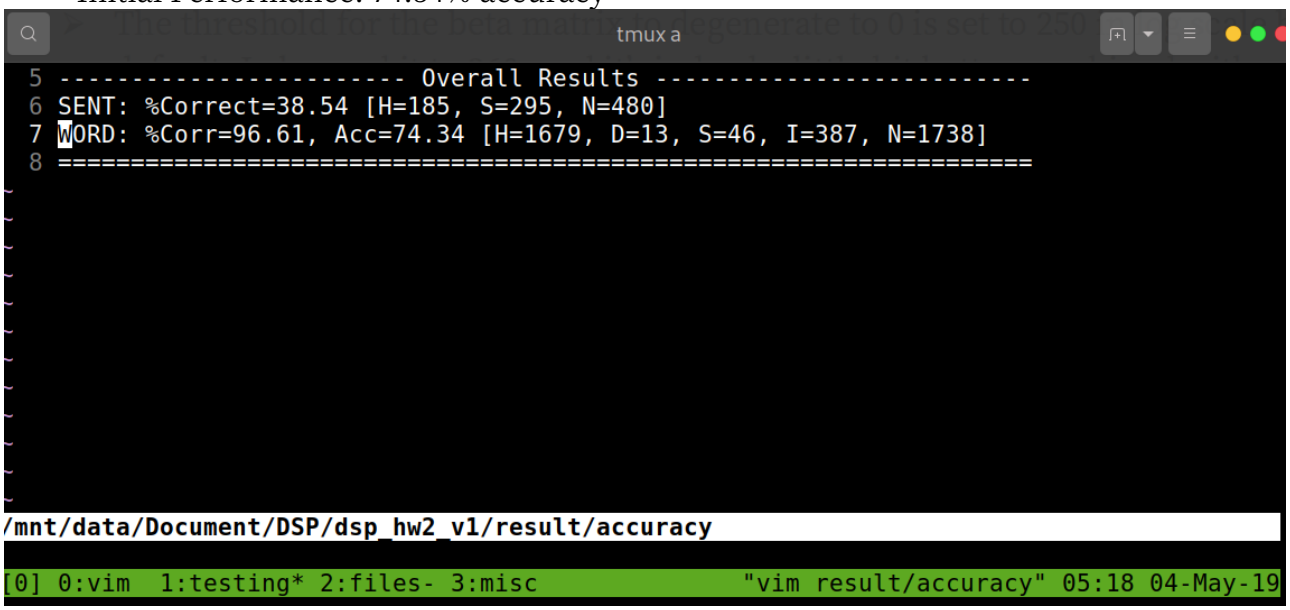
- According to this[1] paper, adjusting the HMM time window ("lib/config.cfg", "lib/hcopy.cfg") of the MFCC to about 15 to 35 ms would yield better results, and my experiments suggest 20ms window is better than 32ms by a slight margin.

- The "sp" hmm have only 1 effective hidden state by default (3 states but first and third are non-emitting). Also, other models (1~9, "sil") have 3 effective states only, too, which is also on the smaller side. Adding more state to both yields considerable improvements; below are some experiment data:
  ("sp" have 2 effective state, while other HMMs have 4): 95.51%
  ("sp" have 3 effective state, while other HMMs have 4): 96.14%

- The default mean and variances are set to standard normal distribution, while the results averages to have variance around 10. Modifying the initial variance ("lib/proto") to around 10 also yields better performance.

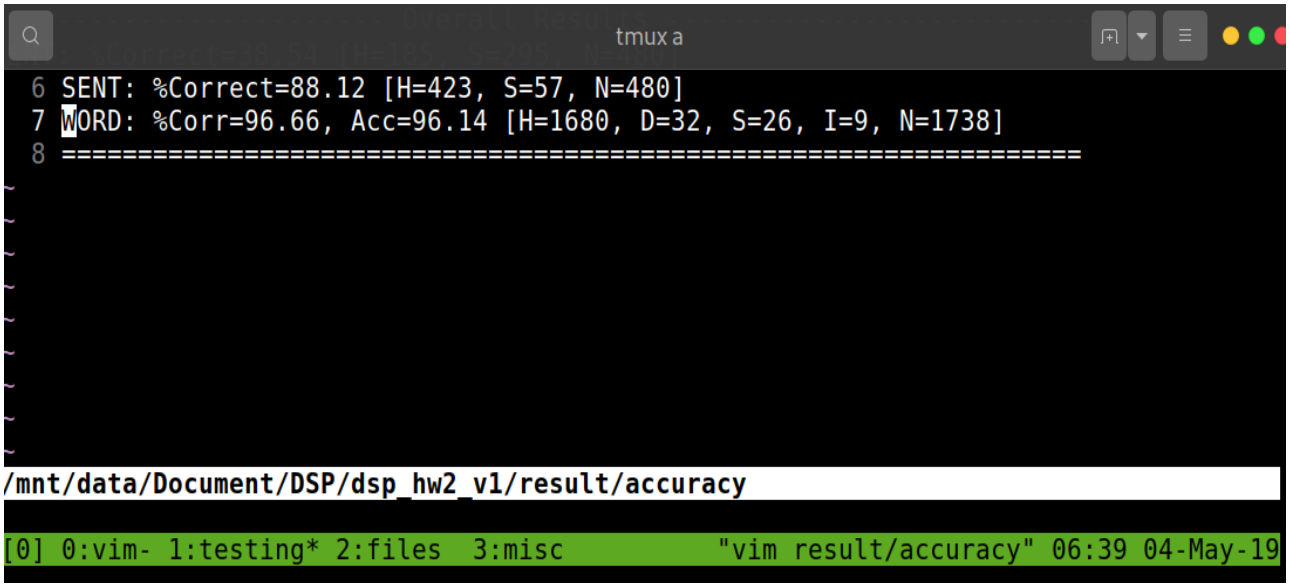
- The threshold for entries in beta matrix to degenerate to 0 is set to 250 in log scale by default. I changed it to 260, and it's indeed a tiny bit better combined with some other parameter adjustments.

- Initial Performance: 74.34% accuracy



- My best performance: 96.20% accuracy

1 "Preference for 20-40 ms window duration in speech analysis"
Kuldip K. Paliwal, James G. Lyons and Kamil K. Wojcicki
Signal Processing Laboratory Griffith University, Nathan, QLD 4111, Australia
{k.paliwal, j.lyons, k.wojcicki}@griffith.edu.au
https://core.ac.uk/download/pdf/143866920.pdf



- Some Thoughts on this homework:
  - HTK is powerful in that we could theoretically build a super sophisticated model – that the manual for the toolkit is several hundreds pages is itself a good illustration - to approximate what we want to model to our hearts content, but that the toolkit have only 32-bit version probably means that the project is not properly maintained for some days. In terms of data-crunching 64-bit variants would definitely be more powerful. Also by monitoring CPU usage, the project isn't well-parallelized either.

  - Since lack of time, I didn't record every resulting change corresponding to a parameter change; all I done in this work is basically increase the information in a hidden state except that I intentionally split the MFCC into 3 streams. There shall be more to discover within these parameter changes.

  - The provided C files are just plain garbage. Command line tools such as awk, sed, etc would do these copy-pasting better in that it would take just several lines of bash script and it would be easier to understand, and that what the comments says the code is doing differs from what the code actually do is simply unacceptable.

- How to run:
  Ensure your system have "bash" and "gcc" and "g++" and various 32-bit binaries needed for HTK.
  If above is met, just type "chmod u+x ./run.sh; ./run.sh" in the command line.