A3 Tutorial 4

- Levenshtein initialization correction
- Log-likelihood calculation for GMMS
- How to verify your GMM
Levenshtein initialization

- Initialize $\text{lev}(i,0)$ to $i$ and $\text{lev}(0,j)$ to $j$, not infinity
- Initializing to infinity does not correctly handle insertion errors at the first word

\[
\text{lev}_{a,b}(i, j) = \begin{cases} 
\max(i, j) & \text{if } \min(i, j) = 0, \\
\min \begin{cases} 
\text{lev}_{a,b}(i - 1, j) + 1 \\
\text{lev}_{a,b}(i, j - 1) + 1 \\
\text{lev}_{a,b}(i - 1, j - 1) + 1_{(a_i \neq b_j)}
\end{cases} & \text{otherwise.}
\end{cases}
\]
Log-likelihood calculation for GMMs

\[
\log b_m(\vec{x}_t) = -\sum_{n=1}^{d} \frac{(\vec{x}_t[n] - \mu_m[n])^2}{2\sigma_m^2[n]} - \frac{d}{2} \log 2\pi - \frac{1}{2} \log \prod_{n=1}^{d} \sigma_m^2[n]
\]

- \( \vec{x}_t \) = a vector of features, \( t \) = index of the vector
- \( d \) = # of features = 14
- \( n \) = the index of the current feature (1-14)
- \( m \) = the index of the Gaussian component (1-8)
- \( \mu_m \) = the mean vector of the mth Gaussian
- \( \sigma_m^2 \) = co-variance vector of the mth Gaussian
- Use the log probability to prevent underflow
- Represent the diagonal of the covariance matrix as a d-dimensional vector
Log-likelihood calculation for GMMs

- This is the SAME equation
- Compute the second part once for each m and store it
- Compute the first part for each feature vector $\mathbf{x}$

\[
\log b_m(\mathbf{x}_t) = - \sum_{n=1}^{d} \left( \frac{1}{2} \mathbf{x}_t[n]^2 \sigma_m^{-2}[n] - \mathbf{\mu}_m[n] \mathbf{x}_t[n] \sigma_m^{-2}[n] \right) \\
- \left( \sum_{n=1}^{d} \frac{\mathbf{\mu}_m[n]^2}{2\sigma_m^2[n]} + \frac{d}{2} \log 2\pi + \frac{1}{2} \log \prod_{n=1}^{d} \sigma_m^{-2}[n] \right)
\]
Some other notes

- We’re using the natural log, or \( \ln \), which is just log in Matlab

- Each training iteration loops over \( t=1 \) to \( T \), which is all of the mfcc vectors for the given speaker, across all utterances

- For the HMM part, the training vectors are all occurrences of that phoneme across all utterances from all speakers
How do I know that my GMM model is good?

- Your probabilities should converge: they should change a lot during earlier iterations, but change less over time. The model has converged when the parameter values stop changing. The parameter $\epsilon$ tells us what difference value counts as “not changing much”. You get to pick this parameter.

- The log probabilities output by your model should be in the -10K range.

- Each component $m$ should have a different mean and weight.

- Your prediction accuracy should be better than chance (hopefully much better).

- If you’re not getting good results, try changing your parameters.
How do I know that my GMM model is good?

- Synthetic dataset (courtesy of a classmate):
  https://gist.github.com/anonymous/d6ad6b1c5a14025e7286209b4659e604