Επεξεργασία Φωνής και Φυσικής Γλώσσας

2ο Εργαστήριο: Αναγνώριση Φωνής με το KALDI TOOLKIT

ΣΧΟΛΗ: ΗΜΜΥ

Ονοματεπώνυμο	Αριθμός Μητρώου
Γιάννης Πιτόσκας	03115077
Αντώνης Παπαοικονόμου	03115140

Σκοπός της άσκησης αυτής είναι η υλοποίηση ενός συστήματος επεξεργασίας και αναγνώρισης φωνής με το εργαλείο Kaldi.

Θεωρητικό υπόβαθρο

Mel-frequency Cepstral Coefficients(MFCCs):

Η διαδικασία της εκπαίδευσης ενός συστήματος αναγνώρισης φωνης, περνάει μέσα από την εξαγωγή δεδομένων χαρακτηριστικών. Η πιο διαδεδομένη μορφή απεικόνισης αυτών είναι ένα MFCC Vector 39 θέσεων. Η ανάπτυξη της ιδέας αυτής χρησιμοποιεί και ιδέες από cepstrum, καθώς και ψυχοακουστικές μελέτες για την μετάφραση του ήχου στην κατάλληλη μορφή . Η εξαγωγή των χαρακτηριστικών τμηματοποιείται σε 6 στάδια:

1) Προέμφαση του ήχου μέσα από την μετατόπιση κάθε φωνήματος σσε υψηλότερες συχνότητες που βοηθούν την αναγνώριση.
2) Παραθυροποίηση των ηχητικών δεδομένων για να διατηρήσουμε την στατικότητα των στατιστικών χαρακτηριστικών του ήχου(μέση τιμή,διακύμανση) και κατεπέκταση την εργοδικότητα(αφού έχουμε πλήθος δεδομένων στο χρόνο αλλά δε μπορούμε να επαναλάβουμε διαδοχικά τις ίδιες εκτελέσεις των ίδιων δεδομένων).Συχνά χρησιμοποιούμε Kaiser παράθυρα με δεδομένο overlap για να ομαλοποιηθούν φαινόμενα των άκρων. 3)
3) Συχνοτική ανάλυση σήματος ,μέσω τηης χρήσης DFT για τον υπολογισμό της συγκέντρωσης της ενέργειας σε κάθε συχνότητα.
4) Mel filter bank διαμόρφωση,που αιτιολογείται επειδή ο ανθρώπινος εγκέφαλος παρουσιάζει δυσκολία στον διαχωρισμό συχνοτήτων σε υψηλοσυχνοτικά σήματα.Γενικά η ανθρώπινη απόκριση είναι γραμμική κάτω από τα 1000Hz και λογαριθμική παραπάνω,οπότε έτσι σχεδιάονται και τα φίλτρα μας.
5) Χρήση cepstrum,για την μετάθεση του σήματος σε πεδίο quefrency και το διαχωρισμό πηγής και φίλτρου.Αποδεικνύεται ότι ο διαχωρισμός αυτός βελτιώνει το μονελο καθώς ο εντοπισμός π.χ. του φίλτρου: Θέση της γλώσσας μπορεί να προσφέρει στην ανάλυση παραπάνω χαρακτηριστικά για την αναγνώριση των φωνημάτων.
6) Delta-Training: Τα χαρακτηριστικά αυτά είναι τιμές που δείχνουν πόσο μεταβάλλονται κάποια στατιστικά χαρακτηριστικά του cepstrum μέσα σε ένα παράθυρο.Συμπληρώνουν το διάνυσμα που δημιουργούμε.Για ακρίβεια χρησιμοποιούμε και τα double-deltas που δείχνουν την μεταβολή των deltas . Αυτός ο χειρισμός του σήματος καταλήγει σε 39 MFC Coeficients και περιέχει το μεγαλύτερο μέρος της πληροφορίας για κάθε παραθυροποιημένο σήμα.

Γλωσσικό Μοντέλο (Language Model):

Για το μοντέλο αυτό συνήθως χρησιμοποιούμε 4-grams ή tri-grams και πιό σπάνια unigrams και bigrams. Ωστόσο, κατά την χρησιμοποίηση του Kaldi για την δημιουργία του μοντέλου αναγνώρισης φωνής βλέπουμε ότι δηιουργεί unigram, bigram και trigram συνδυάζοντάς τα και προφανώς αυξάνοντας την υπολογιστική πολυπλοκότητα αλλά και την ακρίβεια όσο αυξάνεται το μέγεθος του μοντέλου.

Ακουστικό Μοντέλο (Acoustic Model):

Στο στάδιο της δημιουργίας του ακουστικού μοντέλου ουσιαστικά υπολογίζουμε την πιο πιθανή ακολουθία από παρατηρήσεις όταν μας έχουν δοθεί άλλα γλωσσικά χαρακτηριστικά όπως (λέξεις, φωνήματα ή κάποια μέρη φωνημάτων). Πιο συγκεκριμένα δοθέντος μίας πιθανότητας $P(O|W)$ μπορούμε να κατατάξουμε για κάθε HMM μίας κατάστασης q του αυτομάτου μας, χρησιμοποιώντας Gaussian Mixture Models (GMM's) και τα υπόλοιπα γλωσσικά χαρακτηριστικά, την πιθανοφάνεια.

Προπαρασκευή/Κύριο Μέρος

Χωρισμός Δεδομένων:

Μέσα στον φάκελο egs δημιουργούμε τον φάκελο usc μέσα στον οποίο θα εργαζόμαστε από εδώ και στο εξής. Για τον διαχωρισμό των δεδομένων μας τα χωρίζουμε σε 3 sets = {train, dev, test} και έτσι φτιάχνουμε μέσα στον usc και τα αντίστοιχα directories data/train , data/dev , data/test . Μέσα σε καθέναν από αυτούς τους φακέλους δημιουργούμε το εξής σύνολο αρχείων:

- uttids
- utt2spk
- spk2utt
- wav.scp
- text

Τα παραπάνω αρχεία περιγράφουν τις αντιστοιχίσεις που ζητούνται στην εκφώνηση και έχουν τα αντίστοιχα formats.
Στη συνέχεια εφαρμόζουμε ένα preprocessing στις προτάσεις του εκάστοτε αρχείου text και ύστερα, με χρήση του lexicon.txt, αντικαθιστούμε κάθε μία από αυτές τις προτάσεις με την σειρά από φωνήματα που της αντιστοιχεί, πάντα με το φώνημα της σιωπής (sil) στην αρχή και το τέλος της.

Προετοιμασία διαδικασίας αναγνώρισης φωνής για τη USC-TIMIT:

- Από τον φάκελο wsj παίρνουμε το αρχείο path.sh και θέτουμε ως KALDI_ROOT το directory στο οποίο βρίσκεται ο κύριος φάκελος εγκατάστασης του Kaldi.
- Από τον φάκελο wsj παίρνουμε το αρχείο cmd.sh και θέτουμε στις μεταβλητές train_cmd, decode_cmd, cude_cmd τιμή run.pl
- Δημιουργούμε soft/symbolic links μέσα στο φάκελο της δικής σας διαδικασίας με ονόματα 'steps' και 'utils' τα οποία θα δείχνουν στους αντίστοιχους φακέλους της wsj .
- Δημιουργούμε τον φάκελο local και μέσα σε αυτόν ένα soft link που να δείχνει στο αρχείο score_kaldi.sh που βρίσκεται μέσα στο 'steps'.
- Δημιουργείστε το φάκελο conf και μέσα σε αυτόν αντιγράψτε το αρχείο mfcc.conf που σας δώθηκε στις διευκρινίσεις.
- Δημιουργούμε τα directories data/lang, data/local/dict, data/local/lm_tmp, data/local/nist_lm.

Προετοιμασία γλωσσικού μοντέλου:

Μέσα στο directory data/local/dict δημιουργούμε τα παρακάτων αρχεία:

- silence_phones.txt και optimal_phones.txt που περιέχουν μόνο το φώνημα της σιωπής (sil).
- nonsilence_phones.txt με όλα φωνήματα που εμφανίζονται (εκτός από αυτό της σιωπής) και μάλιστα ταξινομημένα.
- lexicon.txt που περιγράφει 1-1 αντιστοιχία κάθε φωνήματος με τον εαυτό του.
- lm_train.text, lm_dev.text, lm_test.text τα οποία προκύπτουν από τα αντίστοιχα αρχεία ονόματος text που φτιάξαμε κατά τον διαχωρισμό των δεδομένων με την προσθήκη των ειδικών μονάδων <s> και </s> στην αρχή και στο τέλος κάθε πρότασης.
- extra_questions.txt το οποίο είναι κενό.

Έχοντας εγκαταστήσει το πακέτο IRSTLM, εκτελώντας extras/install_irstlm.sh στον φάκελο tools του Kaldi, μπορώ να χρησιμοποιώ τις αντίστοιχες εντολές:

- Χρησιμοποιούμε την εντολή build-lm.sh και έτσι δημιουργούμε στο directory data/local/lm_tmp τα built_lm_unigram.ilm.gz και built_lm_bigram.ilm.gz για το unigram και το bigram γλωσσικό μοντέλο αντίστοιχα, που αποτελούν την ενδιάμεση μορφή του γλωσσικού μοντέλου.
- Χρησιμοποιούμε την εντολή compile-lm.sh και έτσι δημιουργούμε στο directory data/local/nist_lm τα compiled_lm_unigram.arpa και compiled_lm_unigram.arpa για το unigram και το bigram γλωσσικό μοντέλο αντίστοιχα, που αποτελούν το compiled γλωσσικό μοντέλο σε μορφή ARPA.
- Στη συνέχεια χρησιμοποιούμε την εντολή prepare_lang.sh για την δημιουργία του αρχείου L.fst στον φάκελο data/lang, το οποίο είναι το FST του λεξικού της γλώσσας.
- Τέλος, ακολουθώντας την λογική της διαδικασίας timit του Kaldi, δημιουργούμε το FST της γραμματικής G.fst, χρησιμοποιώντας τα αρχεία μορφής ARPA που δημιουργήσαμε προηγουμένως.

Ερώτημα 1: Για τα γλωσσικά μοντέλα που δημιουργήσατε υπολογίστε το perplexity στο validation και στο test set. Τι δείχνουν αυτές οι τιμές?

Τρέχοντας το bash script perplexity.sh που έχουμε φτιάξει παίρνουμε το perplexity των sets χρησιμοποιώντας την compile-lm του IRSTLM.

Dev unigram perplexity: % $Nw = 6540$ $PP = 40.13$ $PPwp = 0.00$ $Nbo = 0$ $Noov = 0$ $OOV = 0.00\%$
Dev bigram perplexity: % $Nw = 6540$ $PP = 18.24$ $PPwp = 0.00$ $Nbo = 182$ $Noov = 0$ $OOV = 0.00\%$
Test unigram perplexity: % $Nw = 6363$ $PP = 39.75$ $PPwp = 0.00$ $Nbo = 0$ $Noov = 0$ $OOV = 0.00\%$
Test bigram perplexity: % $Nw = 6363$ $PP = 17.61$ $PPwp = 0.00$ $Nbo = 183$ $Noov = 0$ $OOV = 0.00\%$

Εξαγωγή ακουστικών χαρακτηριστικών:

Χρησιμοποιούμε αρχικά την εντολή fix_data_dir.sh για κάθε ένα από τα 3 sets και έχω τα δεδομένα μου sorted και γενικότερα στην ακριβή μορφή που τα χρειάζονται οι επόμενες εντολές. Ύστερα, χρησιμοποιούμε τις εντολές make_mfcc.sh και compute_cmvn_stats.sh για την εξαγωγή των MFCCs.

Ερώτημα 2: Με τη δεύτερη εντολή πραγματοποιείται το λεγόμενο Cepstral Mean and Variance Normalization. Τι σκοπό εξυπηρετεί? (Bonus: Δώστε μια μαθηματικά τεκμηριωμένη απάντηση)

Στην αναγνώριση φωνής επιθυμούμε γενικά να αφαιρούμε τις επιδράσεις που έχει στο ηχητικό σήμα το εκάστοτε κανάλι διέλευσης. Έστω $x[n]$ το σήμα εισόδου από κάποιον ομιλητή, και έστω μια κρουστική απόκριση $h[n]$ που δημιουργείται από το κανάλι. Τότε το σήμα $y[n]$ που καταγράφεται είναι η έξοδος ενός συστήματος με είσοδο το $x[n]$ και κρουστική απόκριση το h[n], δηλαδή $y[n] = h[n] * x[n]$. Κάνοντας χρήση DFT παίρνουμε $Y[f] = H[f] \cdot X[f]$ καθώς συνέλιξη στο (διακριτό) χρόνο αντιστοιχεί σε πολλαπλασιασμό στο πεδίο των (διακριτών) συχνοτήτων. Στη συνέχεια για τον υπολογισμό του Cepstrum λογαριθμίζουμε την σχέση που έχουμε στο πεδίο των συχνοτήτων:

$$Y_C[q] = \log(Y[f]) = \log(H[f] \cdot X[f]) = \log(H[f]) + \log(X[f]) = H_C[q] + X_C[q]\,,$$

όπου q = quefrency

Αυτό που μπορεί λοιπόν να παρατηρηθεί είναι ότι κάθε παραμόρφωση του σήματος εισόδου, μέσω συνέλιξης μπορεί να αναπαρασταθεί με πρόσθεση στο πεδίο του Cepstrum. Μπορούμε να παρατηρήσουμε ότι για κάθε i-οστό frame ισχύει ότι:

$$Y_{C,i}[q] = H_C[q] + X_{C,i}[q]$$

Παίρνοντας την μέση τιμή πάνω σε όλα τα frames παίρνουμε:

$$\frac{1}{N}\sum_i Y_{C,i}[q] = \frac{1}{N}\sum_i H_C[q] + X_{C,i}[q] \Rightarrow$$
$$\Rightarrow \boldsymbol{\frac{1}{N}\sum_i Y_{C,i}[q] = H_C[q] + \frac{1}{N}\sum_i X_{C,i}[q]}$$

Έστω: $R_{C,i}[q] = Y_{C,i}[q] - \frac{1}{N}\sum_j Y_{C,j}[q] \Rightarrow$

$$\Rightarrow R_{C,i}[q] = H_C[q] + X_{C,i}[q] - H_C[q] - \frac{1}{N}\sum_i X_{C,i}[q] \Rightarrow$$
$$\Rightarrow \boldsymbol{R_{C,i}[q] = X_{C,i}[q] - \frac{1}{N}\sum_i X_{C,i}[q]}$$

Άρα υπολογίζουμε Cepstrum του καταγραφόμενου σήματος, αφαιρούμε την μέση τιμή του από κάθε συνιστώσα του, και τέλος μπορούμε και να διαιρέσουμε με την αντίστοιχη διακύμανση για να πάρουμε το Cepstral Mean Variance Normalization. Έτσι βλέπουμε ότι καταλήγουμε σε κάτι στο οποίο δεν επιδρούν με κανέναν τρόπο οι αλλοιώσεις του καναλιού που αρχικά περιγράψαμε.

Ωστόσο, αυτό δεν είναι ενδεικτικό, καθώς υπάρχουν και περιπτώσεις, όπως όταν έχουμε additive noise, που μπορεί να μας δώσει πολύ λάθος αποτελέσματα:

$$y[n] = h[n] * x[n] + w[n] \Rightarrow$$
$$Y[f] = H[f] \cdot X[f] + W[f] \Rightarrow$$
$$Y_C[q] = \log(Y[f]) \Rightarrow$$
$$Y_C[q] = \log(H[f] \cdot X[f] + W[f]) \Rightarrow$$
$$Y_C[q] = \log\left(X[f] \cdot \left(H[f] + \frac{W[f]}{X[f]}\right)\right) \Rightarrow$$
$$Y_C[q] = \log X[f] + \log\left(H[f] + \frac{W[f]}{X[f]}\right)$$

Σε περιπτώσεις λοιπόν με κακό SNR ο όρος $\frac{W[f]}{X[f]}$ είναι πολύ πιθανό να μας αποπροσανατολίσει κυριαρχόντας κατά την εκτίμηση των αποτελεσμάτων.

Στην γενική περίπτωση το CMVN είναι μια υπολογιστικά αποδοτικη τεχνική για την αναγνώριση φωνής η οποία δίνει μάλιστα πολύ καλα αποτελέσματα. Βέβαια αυτό δεν είναι δεσμευτικό, καθώς σημειώνεται ότι η απόδοση του CMVN ότι υποβαθμίζεται για σύντομα utterances λόγω ανεπαρκών δεδομένων για την εκτίμηση των παραμέτρων και την απώλεια πληροφοριών που διακρίνονται, καθώς όλα τα utterances αναγκάζονται να έχουν μηδενική μέση και μοναδιαία διακύμανση.

<u>Ερώτημα 3:</u> Πόσα ακουστικά frames εξήχθησαν για κάθε μία από τις 5 πρώτες προτάσεις του training set? Τι διάσταση έχουν τα χαρακτηριστικά?

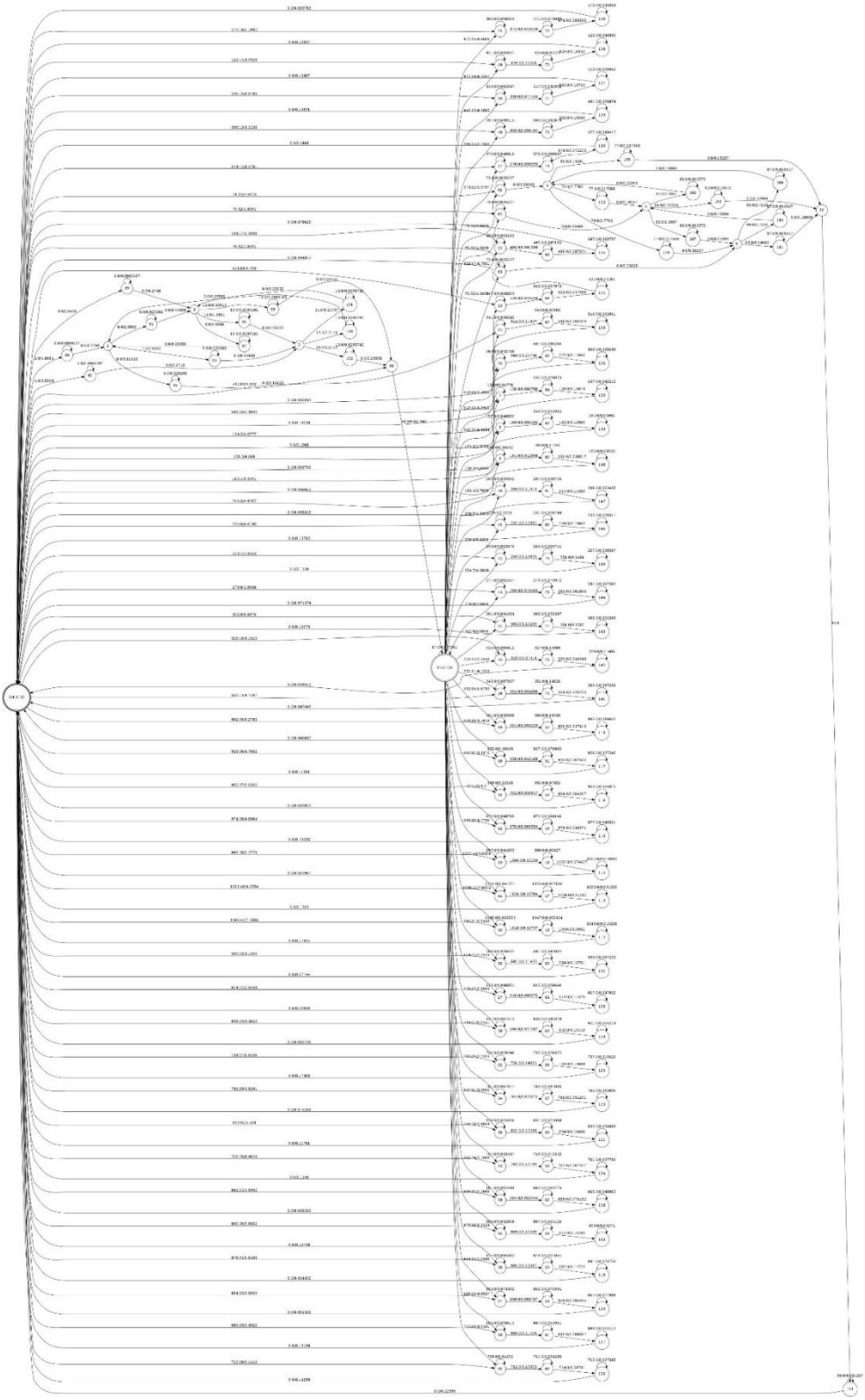
Χρησιμοποιώντας το option --write-utt2num-frames true της make_mfcc.sh παίρνουμε τις παρακάτω τιμές στο αρχείο utt2num-frames για τις πρώτες 5 προτάσεις του train:
f1-STE-1 237
f1-STE-10 306
f1-STE-100 494
f1-STE-101 504
f1-STE-102 461

Εκπαίδευση ακουστικών μοντέλων και αποκωδικοποίηση προτάσεων

Χρησιμοποιώντας το αρχείο train_mono.sh, που βρίσκεται στον υποφάκελο/softlink steps, εκπαιδεύουμε ένα monophone GMM-HMM ακουστικό μοντέλο με τα δεδομένα που έχουμε στον φάκελο data/train με βάση το λεξιλόγιο και την γραμματική που έχουμε δημιουργήσει στα προηγούμενα βήματα.

Για τους γράφους HCLG χρησιμοποιούμε το script utils/mkgraph.sh για τις γραμματικές που έχουμε παραγάγει από πριν στους φακέλους data/lang_test_unigram και data/lang_test_bigram.

Χρησιμοποιώντας την fstdraw μπορούμε να δούμε τον HCLG γράφο για το unigram μοντέλο παρακάτω:



Για το decoding των data/dev και data/test χρησιμοποιούμε το steps/decode.sh, το οποίο υλοποιείται με Viterbi beam pruning. Στο τέλος του decoding καλείται το αρχείο local/score.sh το οποίο και κάνει το scoring του μοντέλου μας και μας επιστρέφει στο directory exp/mono/decode τα Word Error Rates καθώς και τα Sentence Error Rates, για διάφορες τιμές του word insertion penalty. Το WER είναι μια κλασική μετρική για την αξιολόγηση ενός ASR μοντέλου που βασίζεται στην Levenshtein Edit distance και ορίζεται ως:

$$WER = 100\frac{\#del + \#ins + \#sub}{\#total\_words}$$

Σημείωση: Στην δική μας περίπτωση αφού έχουμε αντικαταστήσει τις λέξεις με τα αντίστοιχα φωνήματα από το κομμάτι της προπαρασκευής, έχουμε να κάνουμε με Phoneme Error Rate (PER), με τον τύπο να παραμένει ίδιος με τον παραπάνω απλά με $\#total\_phonemes$ στον παρονομαστή.

Το scoring στο τέλος καλεί το utils/best_wer.pl το οποίο για κάθε μοντέλο παίρνει όλα τα αρχεία τύπου wer_<N> και μας επιστρέφει το μεγαλύτερο WER που αντιστοιχεί σε κάθε μοντέλο. Έτσι έχουμε:

- Για monophone μοντέλο με unigram G:

Test: $\%PER$ 49.46 [2965 / 5995, 52 ins, 1623 del, 1290 sub] $exp/mono/decode\_ug\_test/wer\_7\_0.0$

Validation: $\%PER$ 51.67 [3190 / 6174, 88 ins, 1712 del, 1390 sub] $exp/mono/decode\_ug\_dev/wer\_7\_0.0$

- Για monophone μοντέλο με bigram G:

Test: $\%PER$ 43.80 [2626 / 5995, 95 ins, 1146 del, 1385 sub] $exp/mono/decode\_bg\_test/wer\_7\_0.0$

Validation: $\%PER$ 46.37 [2863 / 6174, 116 ins, 1217 del, 1530 sub] $exp/mono/decode\_bg\_dev/wer\_7\_0.0$

Τέλος παίρνοντας τα alignments που αντιστοιχούν στα αντίστοιχα unigram και bigram monophone μοντέλα με το steps/align_si.sh εκπαιδεύουμε δύο triphone μοντέλα (unigram και bigram) με το αρχείο steps/train_deltas.sh και ακολουθώντας τα ίδια βήματα που περιγράψαμε παραπάνω παίρνουμε τις εξής μετρικές (Θα ασχοληθούμε μόνο με τα test δεδομένα για αυτό το κομμάτι για λόγους απλοποίησης και ταχύτητας):

- Για triphone μοντέλο εκπαιδευμένο με unigram-alignment:

Test: $\%PER$ 36.15 [2167 / 5995, 261 ins, 706 del, 1200 sub] $exp/tri\_ug/decode/wer\_7\_0.0$

- Για triphone μοντέλο εκπαιδευμένο με bigram-alignment:

Test: $\%PER$ 32.46 [1946 / 5995, 235 ins, 545 del, 1166 sub] $exp/tri\_bg/decode/wer\_8\_0.0$

Παρατηρούμε πως όσο αυξάνεται ο «βαθμός» (και η περιπλοκότητα) του μοντέλου τόσο μειώνεται το max Error Rate, με το triphone ακουστικό μοντέλο εκπαιδευμένο με bigram-alignments να έχει την καλύτερη απόδοση.

Σχετικά με τις υπερπαραμέτρους, οι 2 πιο σημαντικές είναι το minumum LM-weight και το maximum LM-weight καθώς και το word insertion penalty. Οι τιμές των παραμέτρων αυτών για το κάθε μοντέλο φαίνονται παραπάνω στο όνομα των WER αρχείων, τα οποία είναι στην μορφή $wer\_ < LM - weight > \_ < word\_ins\_penalty >$.

Ερώτημα 4: Εξηγήστε τη δομή ενός ακουστικού μοντέλου GMM-HMM. Τι σκοπό εξυπηρετούν τα μαρκοβιανά μοντέλα στη συγκεκριμένη περίπτωση και τι τα μίγματα γκαουσιανών? Με ποιό τρόπο γίνεται η εκπαίδευση ενός τέτοιου μοντέλου?

Προκειμένου να κάνουμε speech recognition, θέλουμε να έχουμε έναν ταξινομητή που θα προσδιορίζει ποιο φώνημα (αν υπάρχει) εκφράζεται σε κάθε frame. Θα μπορούσαμε να χρησιμοποιήσουμε ένα απλό GMM για αυτό, δηλαδή π.χ. για κάθε τάξη φωνήματος, να κάνει fit ένα GMM χρησιμοποιώντας όλα τα frames στα οποία βρίσκεται αυτό το φώνημα - για να κάνω classify ένα νέο utterance, θα πρέπει να ελέγξουμε κάθε frame ώστε να βρούμε πιο φώνημα έχει την μεγαλύτερη πιθανότητα εμφάνισης (δηλαδή ποιο GMM δίνει την υψηλότερη πιθανότητα

δημιουργίας του 39 feature vector). Ωστόσο, αυτό το είδος μοντέλου δεν εκμεταλλεύεται τις χρονικές εξαρτήσεις του ακουστικού σήματος - η ταξινόμηση εξαρτάται μόνο από το τρέχον frame, αγνοώντας το context, δηλαδή τα προηγούμενα και τα επόμενα frames. Επιπλέον, αυτό το μοντέλο υποθέτει ότι δεν υπάρχει ακουστική διαφορά στην αρχή, στο μέσο και στο τέλος ενός φωνήματος που δεν είναι σωστό να γίνει μια τέτοια υπόθεση καθώς υπάρχουν φωνήματα που παρότι διαρκούν πολύ λίγο έχουν παρουσιάζουν διακυμάνσεις στον «τόνο» της φωνής.

Το μοντέλο GMM-HMM αποτελεί απάντηση σε αυτά τα προβλήματα. Το HMM είναι ένα χρονικό μοντέλο, το οποίο υποθέτει ότι το source έχει κάποια κατάσταση την οποία δεν γνωρίζουμε (π.χ τοποθέτηση γλώσσας). Οι συνηθέστερες HMM αρχιτεκτονικές για speech recognition έχουν μοντέλα φωνημάτων που αποτελούνται από τρία states. Μπορούμε να κάνουμε την υπόθεση ότι όταν εκφράζεται ένα φώνημα, έχει τρεις διαφορετικές φάσεις - αρχή, μέση και τέλος και σε κάθε φάση, ακούγεται λίγο διαφορετικά. Κάθε μία εκ των καταστάσεων διαμορφώνεται από ένα GMM για να καθορίσει την πιθανότητα της παρατήρησης (observation likelihood) σε αυτή την κατάσταση, και οι παρατηρήσεις μας είναι τα frames.

Συνεπώς έχουμε μια σειρά από frames, καθένα από τα οποία γίνεται classify ως ένα state, και αντιστοιχεί σε κάποιο φώνημα. Μπορεί να υπάρχουν πολλά frames που παράγονται από ένα state και μπορεί να υπάρχουν αρκετά states που αντιστοιχούν σε ένα μόνο φώνημα (και εν συνεχεία, μπορεί να υπάρχουν πολλά φωνήματα που δημιουργούν μια λέξη).

<u>Ερώτημα 5</u>: Γράψτε πώς υπολογίζεται η a posteriori πιθανότητα σύμφωνα με τον τύπο του Bayes για το πρόβλημα της αναγνώρισης φωνής. Συγκεκριμένα, πώς βρίσκεται η πιο πιθανή λέξη (ή φώνημα στην περίπτωσή μας) δεδομένης μίας ακολουθίας ακουστικών χαρακτηριστικών?

Για το πρόβλημα της αναγνώρισης φωνής η ιδέα είναι να υπολογίσουμε την a posteriori πιθανότητα $P(Phoneme_i \mid X)$ σύμφωνα με τον τύπο του Bayes. Ουσιαστικά, αρκεί να υπολογιστεί η εξής παράσταση:

$$P(Phoneme_i \mid X) = \frac{p(X \mid Phoneme_i) \cdot P(Phoneme_i)}{p(X)} \Rightarrow$$

$$\Rightarrow P(Phoneme_i \mid X) = \frac{p(X \mid Phoneme_i) \cdot P(Phoneme_i)}{\sum_{j=1}^{n} p(X \mid Phoneme_j) \cdot P(Phoneme_j)}$$

Όπου:

- Το $P(Phoneme_i)$ αποτελεί την πιθανότητα εμφάνισης του φωνήματος $Phoneme_i$.
- Το $p(X \mid Phoneme_i)$ αποτελεί την πιθανότητα ανίχνευσης της ακολουθίας ακουστικών χαρακτηριστικών X, δεδομένου ότι μέσα σε αυτήν εκφράζεται το φώνημα $Phoneme_i$.
- Το $p(X)$ αποτελεί την πιθανότητα παρατήρησης της ακολουθίας ακουστικών χαρακτηριστικών X.

Προκειμένου να βρούμε το πιο πιθανό φώνημα δεδομένης μιας ακολουθίας ακουστικών χαρακτηριστικών πρέπει να βρούμε το i, δηλαδή ουσιαστικά το $Phoneme_i$ το οποίο μεγιστοποιεί την παραπάνω a posteriori πιθανότητα $P(Phoneme_i \mid X)$, δηλαδή το ζητούμενο είναι το εξής:

$$\max_i\{P(Phoneme_i \mid X)\} = \max_i\left\{\frac{p(X \mid Phoneme_i) \cdot P(Phoneme_i)}{p(X)}\right\}$$

Ωστόσο, ο όρος $p(X)$ είναι απλά ένας όρος που κανονικοποιεί την παραπάνω παράσταση και εμείς δεν επιθυμούμε την μέγιστη a posteriori πιθανότητα, παραμόνο για ποιο φώνημα αυτή επιτυγχάνεται. Επομένως, αρκεί να βρούμε για πιο φώνημα μεγιστοποιείται η παράσταση του αριθμητή $p(X \mid Phoneme_i) \cdot P(Phoneme_i)$. Άρα, τελικά θέλω να βρούμε το i, δηλαδή ουσιαστικά το $Phoneme_i$ το οποίο μεγιστοποιεί την παράσταση $p(X \mid Phoneme_i) \cdot P(Phoneme_i)$, δηλαδή:

$$\max_i\{\, p(X \mid Phoneme_i) \cdot P(Phoneme_i) \,\}$$

Το ζητούμενο ορθά γράφεται ως εξής λοιπόν:

$$\tilde{P}honeme = \underset{i\,:\,Phoneme_i}{arg\max} \{\, p(X \mid Phoneme_i) \cdot P(Phoneme_i) \,\}$$

Ουσιαστικά, η συνάρτηση argmax θα μας δώσει το πιο πιθανό φώνημα δεδομένης της ακολουθίας ακουστικών χαρακτηριστικών X.

Ερώτημα 6: Εξηγήστε τη δομή του γράφου HCLG του Kaldi περιγραφικά.

Αρχικά, να παρουσιάσουμε την συνολική εικόνα για τη δημιουργία γραφημάτων αποκωδικοποίησης, περιγράφοντας τις αντιπροσωπεύει κάθε γράμμα της ονομασίας "HCLG" όπως παρουσιάζεται στο documentation του Kaldi:

- Το G αντιστοιχεί σε έναν αποδοχέα (input symbols = output symbols) που κωδικοποιεί το μοντέλο γραμματικής ή γλώσσας.
- Το L είναι το λεξικό, με output symbols τις λέξεις και input symbols τα φωνήματα.
- Το C αντιπροσωπεύει την εξάρτηση από τα συμφραζόμενα (context-dependency): τα output symbols του είναι τα φωνήματα και τα input symbols του αντιπροσωπεύουν τα context-dependent φωνήματα.
- Το H περιέχει τους ορισμούς HMM. τα output symbols του αντιπροσωπεύουν context-dependent φωνήματα και τα input symbols του είναι transition-ids, τα οποία κωδικοποιούν το pdf-id και άλλες πληροφορίες.

Επομένως, αυτό που μπορούμε να βγάλουμε σαν συμπέρασμα για τον γράφο HCLG είναι ότι πρόκειται για ένα πλήρως αναπτυγμένο γράφημα αποκωδικοποίησης που αντιπροσωπεύει το γλωσσικό μοντέλο, το λεξικό φωνημάτων (lexicon), το context-dependency και τη δομή των HMM στο μοντέλο μας. Το output είναι ένα FST το οποίο έχει word-ids στην έξοδο και pdf-ids στην είσοδο (δείκτες που επιλύονται σε GMMs-Gaussian Mix Models).