Βήμα 1

Με χρήση του προγράμματος Praat πραγματοποιήθηκε ανάλυση στα αρχεία *onetwothree1.wav* και *onetwothree8.wav*. Η μέση τιμή του pitch καθώς και των 3 πρώτων formants για τα φωνήεντα “α”, “ου”, “ι” για τις εκφωνήσεις των ψηφίων “one two three” από δύο ομιλητές (άνδρας και γυναίκα) δίδονται στους παρακάτω πίνακες:

Man					
Digit	**Phonetic**	**Mean pitch**	**F1**	**F2**	**F3**
1	“ου”	133.9 Hz	523.76 Hz	976.99 Hz	2322.42 Hz
1	“α”	134.44 Hz	776.34 Hz	1028.19 Hz	2359.8 Hz
2	“ου”	129.8 Hz	356.58 Hz	1782.21 Hz	2391.74 Hz
3	“ι”	130.52 Hz	386.47 Hz	1898.38 Hz	2328.24 Hz

Woman					
Digit	**Phonetic**	**Mean pitch**	**F1**	**F2**	**F3**
1	“ου”	186.06 Hz	541.56 Hz	858.36 Hz	2356.98 Hz
1	“α”	178.73 Hz	865.7 Hz	1331.86 Hz	2853.40 Hz
2	“ου”	187. 87 Hz	332 Hz	1701.64 Hz	2669.19 Hz
3	“ι”	179.14 Hz	343.73 Hz	2188.34 Hz	2957.55 Hz

Από τις μετρήσεις παρατηρούμε τα εξής:

1. Το mean pitch του άνδρα είναι αισθητά χαμηλότερο, έχει δηλαδή πιο «μπάσα» φωνή.
2. Τα 3 πρώτα formants παρουσιάζουν διαφορές μεταξύ τους, φαίνεται δηλαδή πως αρκούν για να αποφανθούμε ποιο φωνήεν λέει ο εκφωνητής.
3. Οι εκφωνήσεις του “ου” παρουσιάζουν διαφορές μεταξύ τους ως προς τα formants (όχι ως προς το pitch όμως), επομένως παρότι είναι το ίδιο φωνήεν, μπορούμε να ξεχωρίσουμε σε ποιο ψηφίο αντιστοιχούν.

Βήμα 2

Κατασκευάστηκε η συνάρτηση data_parser που δέχεται ως είσοδο:

variable	**type**	**notes**

folder	(str)	Η τοποθεσία των .wav αρχείων

Και δίνει ως έξοδο:

variable	**type**	**notes**
wav_list	List (array)	Οι κυματομορφές των σημάτων διακριτοποιημένες με χρήση της βιβλιοθήκης librosa
speaker_list	List (int)	Το id του ομιλητή για κάθε αρχείο
digit_list	List (str)	Το ψηφίο που εκφωνείται σε κάθε αρχείο
fs	int	Η συχνότητα δειγματοληψίας

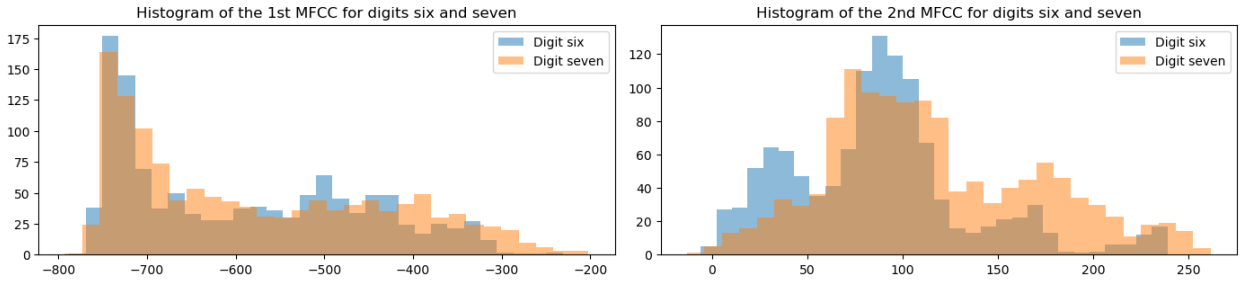
Βήμα 3

Για κάθε αρχείο ήχου και για κάθε timeframe της κυματομορφής εξάγουμε 13 Mel-Frequency Cepstral Coefficients (MFCC), όπως και τις πρώτες και δεύτερες παραγώγους τους χρησιμοποιώντας μήκος παραθύρου 25 ms και βήμα 10 ms.

Βήμα 4

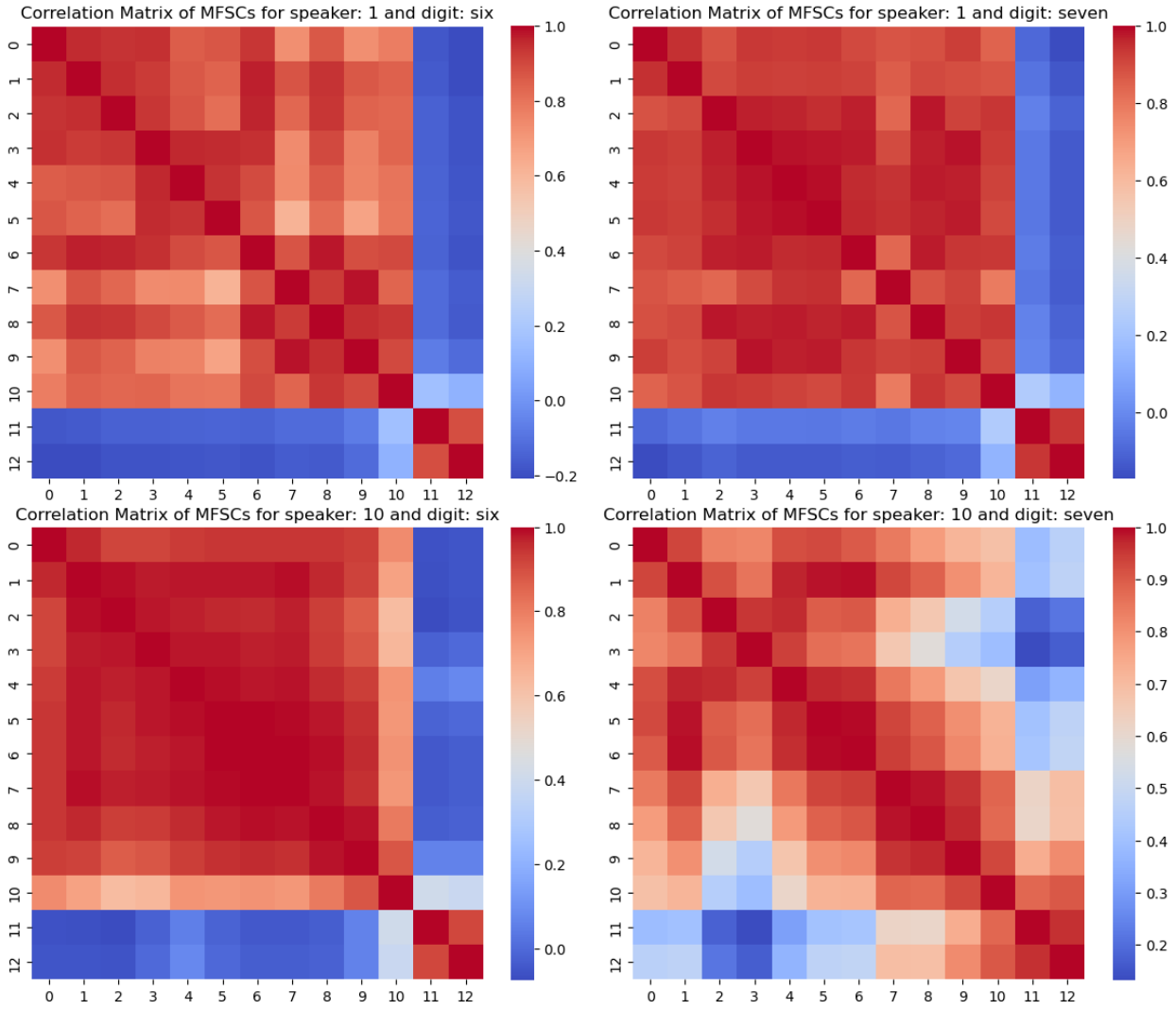
Τα ψηφία που επιλέχθηκαν με βάση τους αριθμούς μητρώου μας είναι τα $n_1 = 6$ και $n_2 = 7$.

Τα ιστογράμματα του $1^{ου}$ και $2^{ου}$ MFCC για όλες τις εκφωνήσεις των ψηφίων φαίνονται παρακάτω:



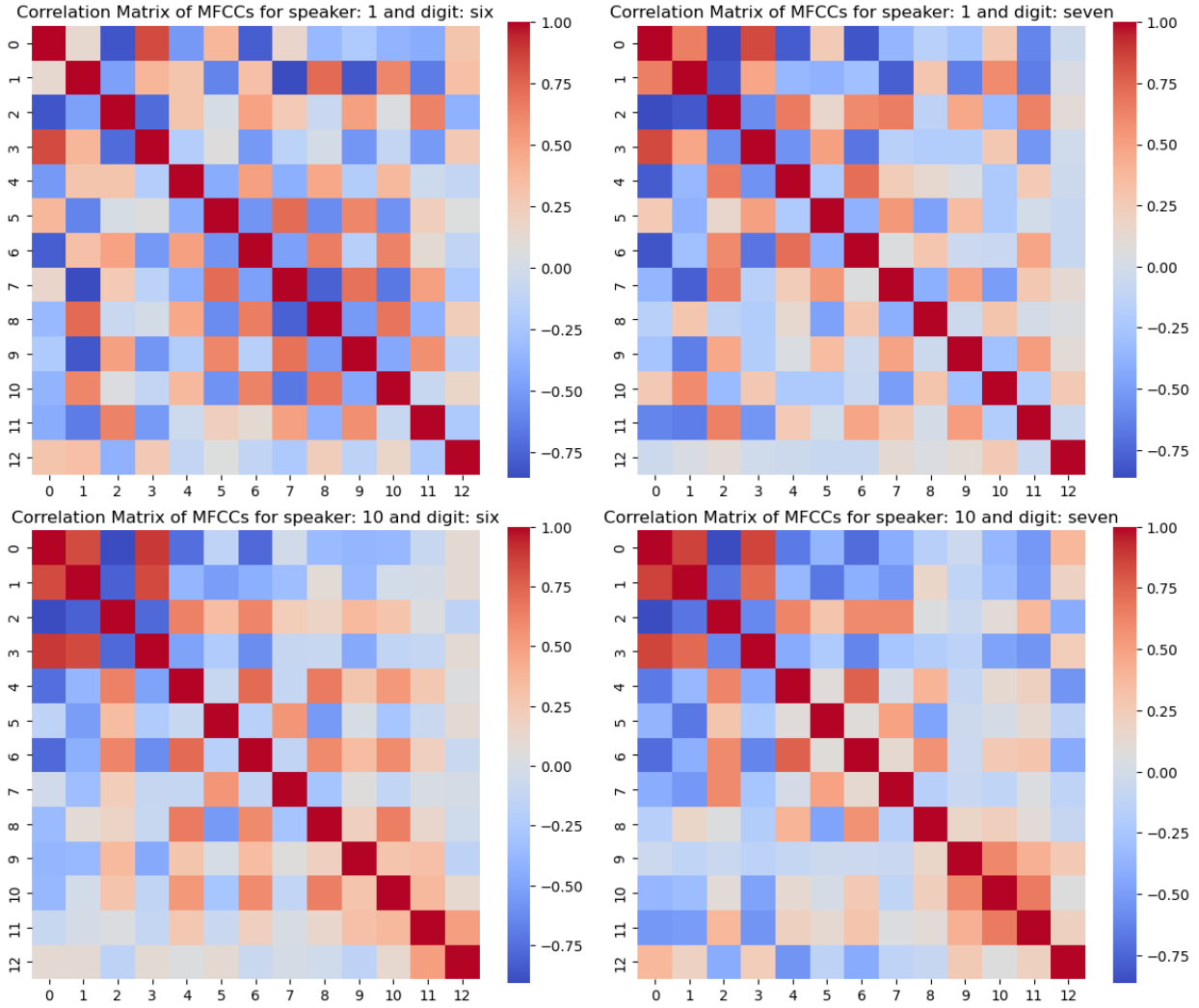
Παρατηρούμε πως υπάρχει μεγάλη επικάλυψη και στα δύο ιστογράμματα, επομένως δεν μπορούμε να ξεχωρίσουμε τα ψηφία με αυτό τον τρόπο.

Η συσχέτιση των Mel Filterbank Spectral Coefficients (MFSCs) για τα ψηφία 6 και 7 παρουσιάζονται στα παρακάτω heatmaps:



Παρατηρούμε πως τα MFSCs έχουν υψηλή θετική συσχέτιση μεταξύ τους.

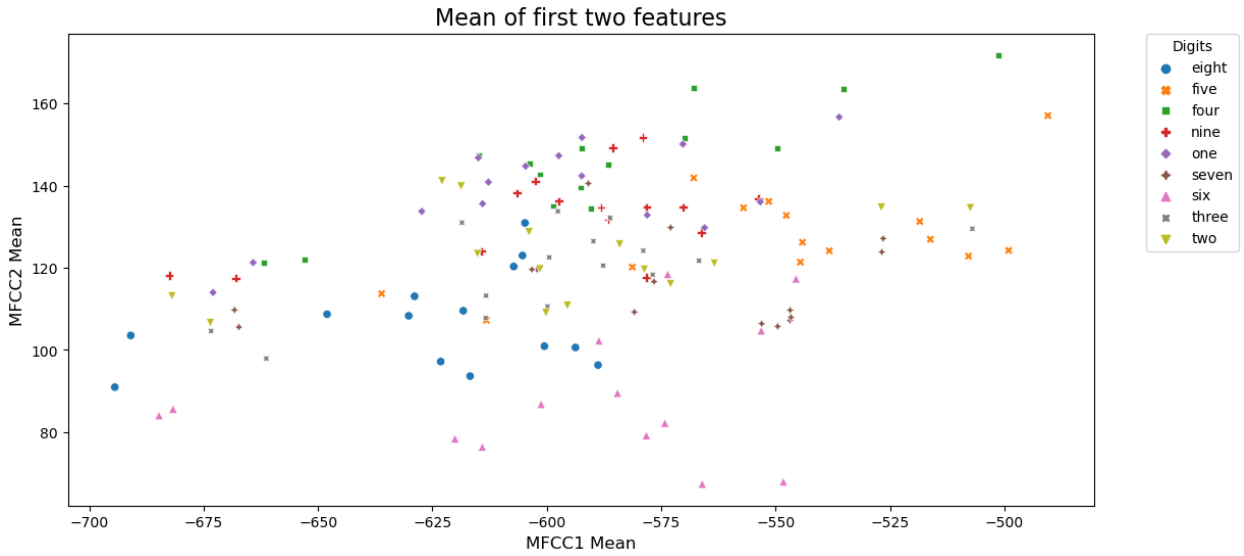
Επαναλαμβάνουμε την ίδια διαδικασία για τα MFCCs:

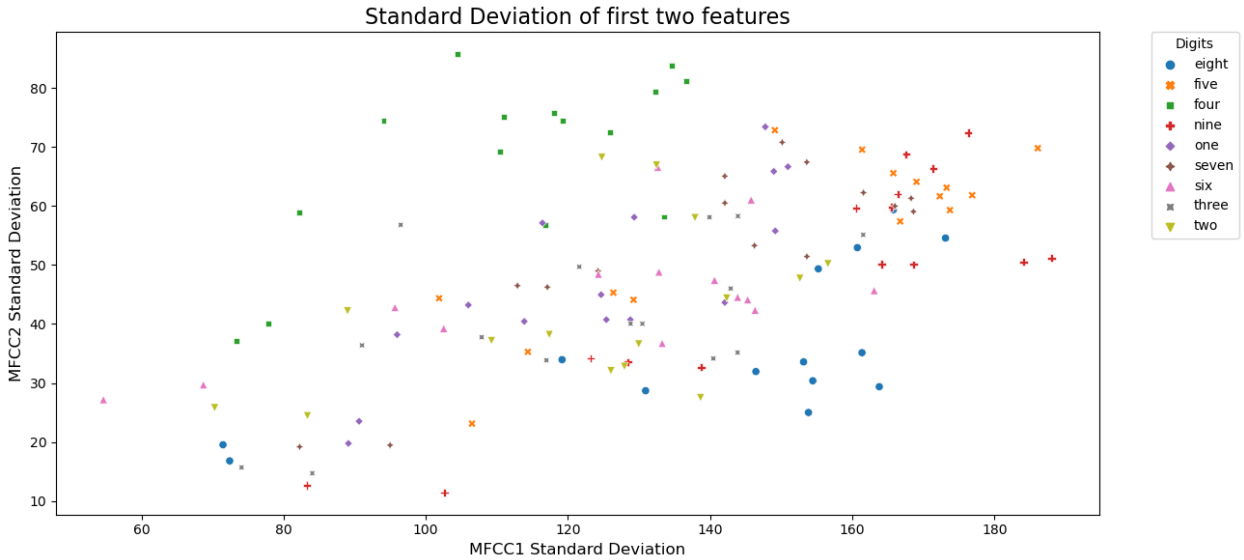


Παρατηρούμε πως για τα MFCCs έχουμε αισθητά μικρότερη συσχέτιση μεταξύ τους. Εξαιρώντας τη διαγώνιο υπάρχουν ελάχιστα ζεύγη MFCC που παρουσιάζουν υψηλή θετική ή αρνητική συσχέτιση. Για αυτό επιλέγουμε τα MFCCs στην επεξεργασία σήματος καθώς έχουν ελάχιστη κοινή πληροφορία.

Βήμα 5

Σχηματίζουμε έναν ενιαίο πίνακα διαστάσεων 39 x #frames για κάθε δείγμα, όπου τα 39 χαρακτηριστικά αποτελούνται από τα 13 MFCCs, τις 13 πρώτες παραγώγους και τις 13 δεύτερες παραγώγους. Στη συνέχεια για κάθε χαρακτηριστικό υπολογίζουμε τη μέση τιμή και την τυπική απόκλιση. Έπειτα αναπαριστούμε με διάγραμμα διασποράς τις μέσες τιμές και τις τυπικές αποκλίσεις των 2 πρώτων MFCC:

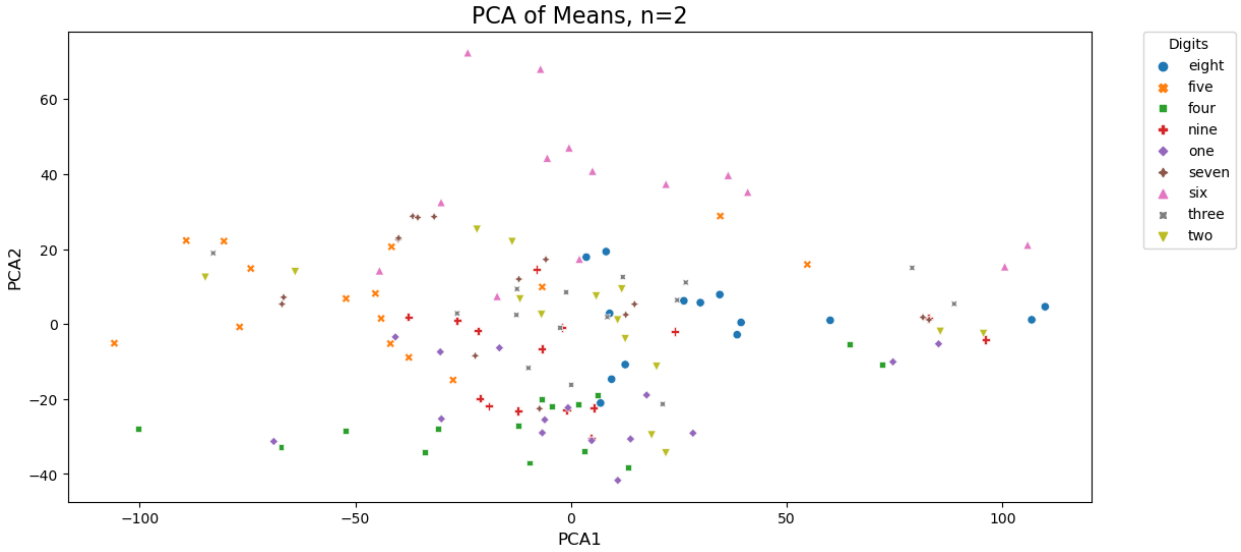


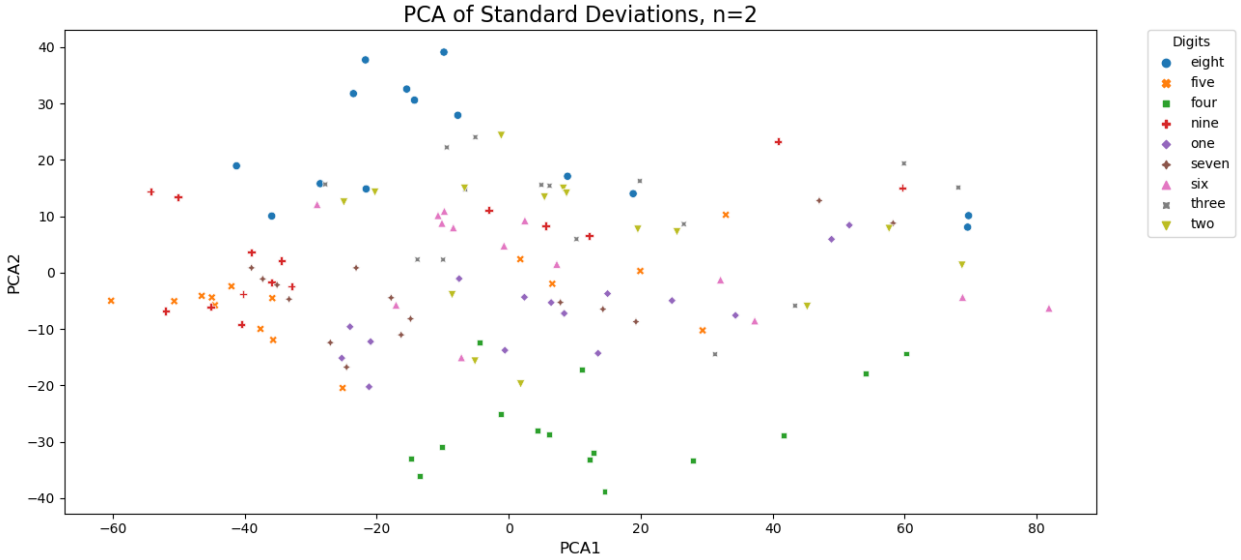


Παρατηρούμε πως με τα υπάρχοντα χαρακτηριστικά δεν μπορούμε να ταξινομήσουμε τα ψηφία.

Βήμα 6

Χρησιμοποιώντας τον αλγόριθμο του Principal Component Analysis (PCA) μειώνουμε τις διαστάσεις των χαρακτηριστικών από 39 σε 2 και αναπαριστούμε με διάγραμμα διασποράς την $1^{η}$ συνιστώσα ως προς τη $2^{η}$ τόσο για τη μέση τιμή όσο και για τη διασπορά:

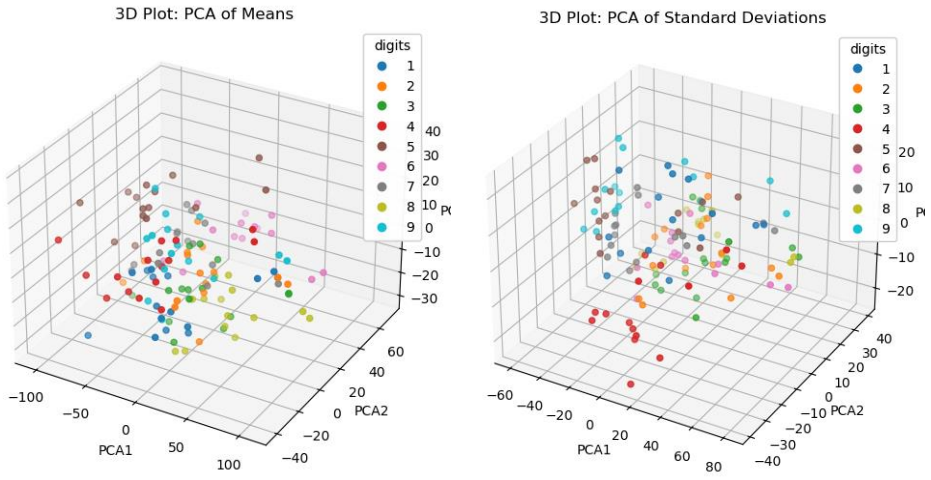





```
Explained variance ratio of the mean from PCA1: 0.670442482218784
Explained variance ratio of the mean from PCA2: 0.15038569322652287
Explained variance ratio of the standard deviation from PCA1: 0.67710384
73889731
Explained variance ratio of the standard deviation from PCA2: 0.16180712
3495478
```

Βλέπουμε ότι διατηρείται το 67% + 15% = 82% της αρχικής διασποράς για τη μέση τιμή και το 68% + 16% = 84% της αρχικής διασποράς για τη τυπική απόκλιση.

Με τον ίδιο αλγόριθμο μειώνουμε από 39 σε 3 διαστάσεις και επαναλάμβανουμε τη διαδικασία:




```
Explained variance ratio of the mean from PCA1: 0.670442482218784
Explained variance ratio of the mean from PCA2: 0.15038569322652287
Explained variance ratio of the mean from PCA3: 0.06208555276099411
Explained variance ratio of the standard deviation from PCA1: 0.6771038473889731
Explained variance ratio of the standard deviation from PCA2: 0.161807123495478
Explained variance ratio of the standard deviation from PCA3: 0.0680939670169802
```

Εδώ διατηρείται το 88% της διασποράς για τη μέση τιμή και το 91% της διασποράς για την τυπική απόκλιση.

Συμπεραίνουμε πως η μείωση σε 2 ή 3 διαστάσεις διατηρεί ικανοποιητικό ποσοστό της πληροφορίας (>80%) επομένως είναι επιτυχημένη.

Βήμα 7

Σε αυτό το βήμα, χωρίζουμε τα δείγματα σε train-test με αναλογία 70-30, πραγματοποιούμε κανονικοποίηση και χρησιμοποιούμε τους εξής ταξινομητές και λαμβάνουμε τα εξής αποτελέσματα:

Classifier	**Test Set Accuracy**
Naive Bayes	0.6000
kNN (k=7)	0.5500
Random Forest (n=100)	0.8000
SVM	0.8000

Οι Random Forest και SVM εμφανίζουν αισθητά μεγαλύτερη επιτυχία ταξινόμησης.

Bonus

Χρησιμοποιώντας τα επιπλέον χαρακτηριστικά:

- Zero Crossing Rate
- Spectral Centroid
- Spectral Rolloff
- Spectral Contrast (7: 1 για όλο το φάσμα συχνοτήτων + 6 για 6 διαφορετικές ζώνες συχνοτήτων)
- Chroma Features (12 για τις 12 διαφορετικές νότες της οκτάβας)
- Spectral Flatness
- Root Mean Square Error

Έτσι καταλήξαμε με 63 χαρακτηριστικά και λάβαμε τα εξής αποτελέσματα:

Classifier	**Test Set Accuracy**
Naive Bayes	0.6250
kNN (k=7)	0.5500
Random Forest (n=100)	0.8750
SVM	0.8750

Παρατηρούμε σχετικά σημαντική αύξηση (7.5%) στην ακρίβεια του Random Forest και του SVM οι οποίοι ήταν ήδη οι πιο ακριβείς στη ταξινόμηση.

Βήμα 8

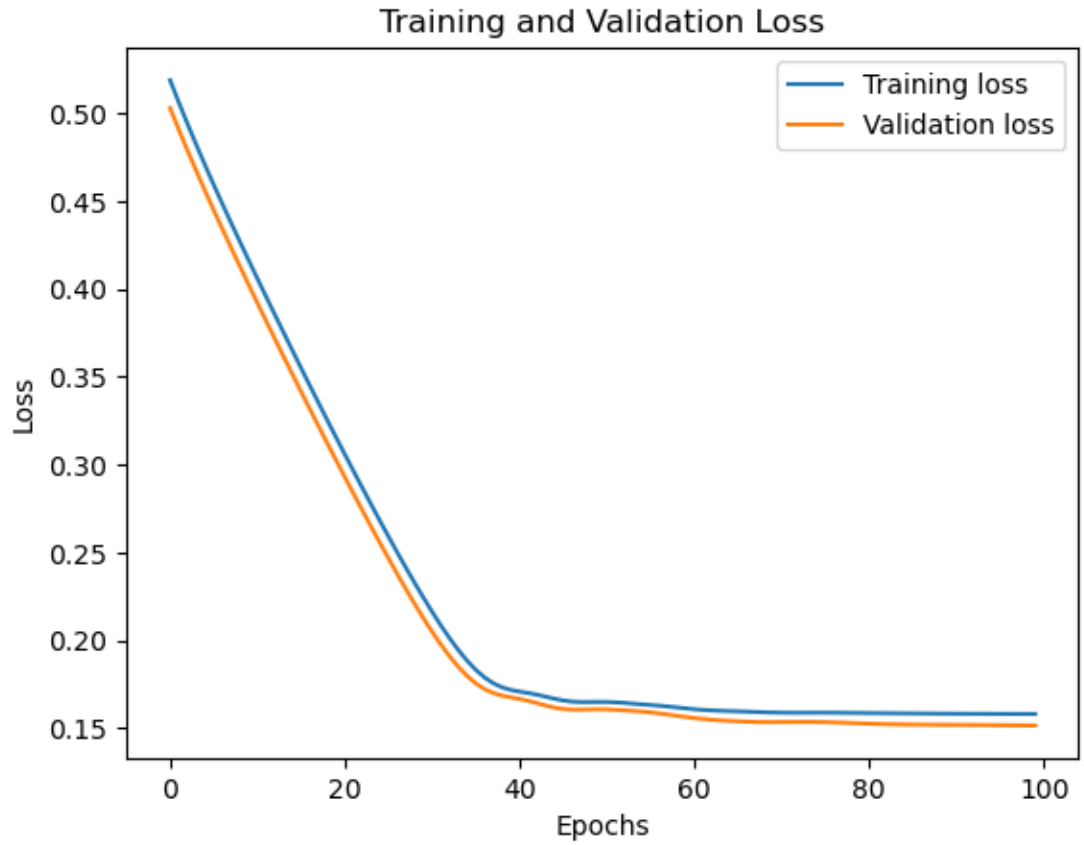
Δημιουργήθηκαν 1000 ακολουθίες 10 σημείων ενός ημιτόνου και ενός συνημιτόνου με A = 1 και f = 40 Hz.

Εκπαιδεύτηκε RNN με 70/30 train/test split και υπερπαραμέτρους:

- Hidden_size = 50
- Num_layers = 1
- Adam optimizer
- Mean Square Error loss
- Learning rate 0.001

Μετά το πέρας της εκπαίδευσης, καταλήγουμε σε validation loss = 0.152 και test loss = 0.168

Τέλος, παρουσιάζουμε το διάγραμμα των validation και train losses ως προς τα epochs



Βήμα 9

Για το βήμα 9 χρησιμοποιήθηκε ο έτοιμος κώδικας του αρχείου parser.py, με εξαίρεση τη συνάρτηση split_free_digits η οποία τροποποιήθηκε ώστε να γυρνά απευθείας χωρισμένα τα δεδομένα των train, dev και test set, με αναλογία 90/10 στο train+dev/test split και 80/20 στο train/dev split. Μεριμνήσαμε να διατηρηθεί σταθερός αριθμός διαφορετικών ψηφίων σε κάθε set.

Εξήχθησαν τα 13 πρώτα MFCCs ως δεδομένα.

Dataset: 3000 .wav αρχεία (10 ψηφία x 6 εκφωνητές x 50 εκφωνήσεις)
Συχνότητα δειγματοληψίας: Fs = 8000 Hz
Μήκος παραθύρου: 240 δείγματα
Βήμα: 120 δείγματα

Μετά το χωρισμό, έχουμε 2160 δείγματα στο train set, 300 στο test set και 540 στο dev set.

Βήματα 10-13

Ορίσαμε τις παραμέτρους για το μοντέλο HMM-GMM:

Πίνακας μετάβασης : $A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 0 & 1 \end{bmatrix}$

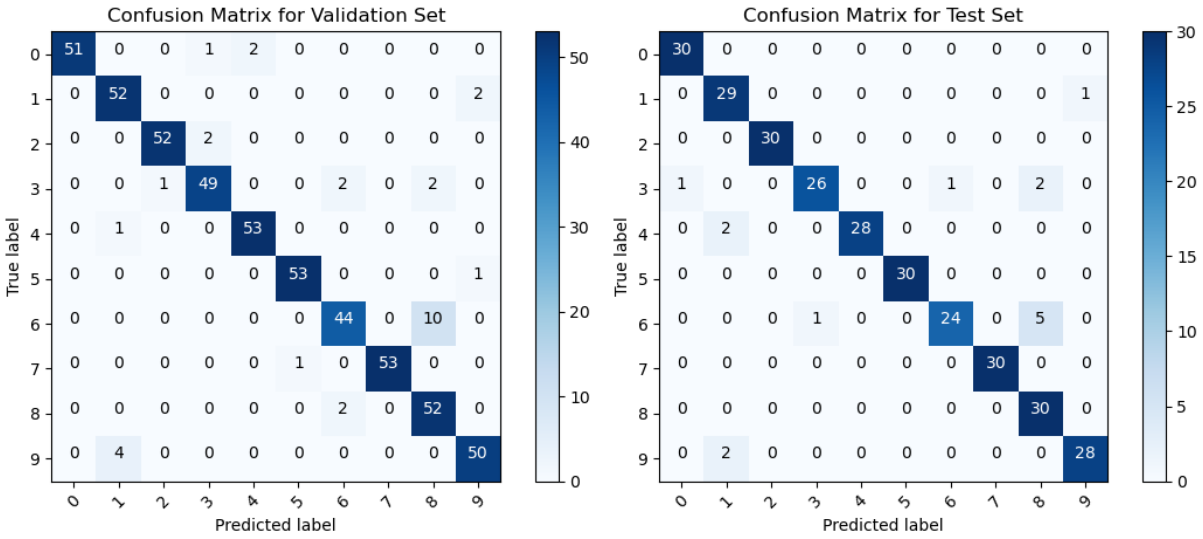
Αρχικές πιθανότητες καταστάσεων: $\Pi_s = [\Pi_1\ \Pi_2\ \Pi_3] = [1\ 0\ 0]$
Τελικές πιθανότητες καταστάσεων: $\Pi_f = [\Pi_1\ \Pi_2\ \Pi_3] = [0\ 0\ 1]$

Τα μοντέλα HMM-GMM θα χρησιμοποιηθούν για αναγνώριση ψηφίων (0-9). Επιλέχθηκαν οι υπερπαράμετροι n_states = 3 και n_mixtures = 2 που είναι ο αριθμός καταστάσεων του HMM και ο αριθμός κανονικών κατανομών που χρησιμοποιούνται για τη δημιουργία του GMM. Με τα states σπάμε τα φωνήματα σε ένα συγκεκριμένο αριθμό καταστάσεων.

Το μοντέλο αρχικοποιήθηκε και εκπαιδεύτηκε μέσω της έτοιμης κλάσης DenseHMM η οποία χρησιμοποιεί τον Expectation Maximization αλγόριθμο για την εκπαίδευση. Τα δεδομένα έγιναν fit στο object του μοντέλου. Η σύγκλιση του μοντέλου γίνεται μέσω της μεταβολής του Log Likelihood.

Για τη διαδικασία επαλήθευσης, από το validation set παίρνουμε ένα διάνυσμα χαρακτηριστικών και με τη μέθοδο log_probablity υπολογίζουμε το λογάριθμο της πιθανότητας το διάνυσμα να ανήκει σε καθένα από τα 10 μοντέλα. Λαμβάνοντας τη μέγιστη τιμή, επιλέγουμε σε ποιο μοντέλο ανήκει, δηλαδή ποιο ψηφίο εκφωνείται.

Στο μοντέλο μας πέτυχαμε Test Accuracy 95%, όπως φαίνεται και από τα Confusion Matrices:



Η εύρεση των βέλτιστων παραμέτρων επιτρέπει στο μοντέλο να εντοπίσει κρυφά μοτίβα για μεγαλύτερη ακρίβεια και να γίνει τόσο σύνθετο όσο χρειάζεται ώστε να μην έχουμε φαινόμενα overfitting/underfitting, δηλαδή η πολυπλοκότητα του μοντέλου να προσαρμοστεί στη πολυπλοκότητα των δεδομένων. Συγκεκριμένα στην αναγνώριση φωνής, είναι σημαντικό το μοντέλο να προσαρμοστεί σε συνθήκες που αλλοιώνουν τα δεδομένα ήχου, όπως θόρυβος.

Επομένως αν δεν πραγματοποιηθεί αυτή η διαδικασία υπάρχει κίνδυνος το μοντέλο να μην μπορεί να εξάγει σωστά συμπεράσματα σε πραγματικά δεδομένα.

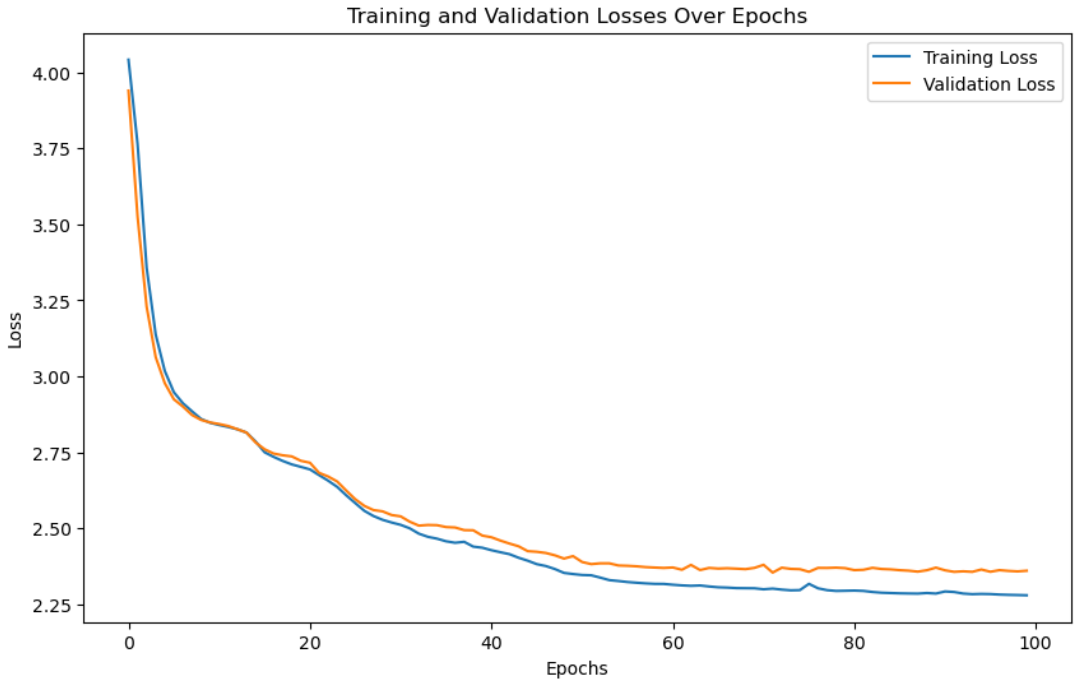
Βήμα 14

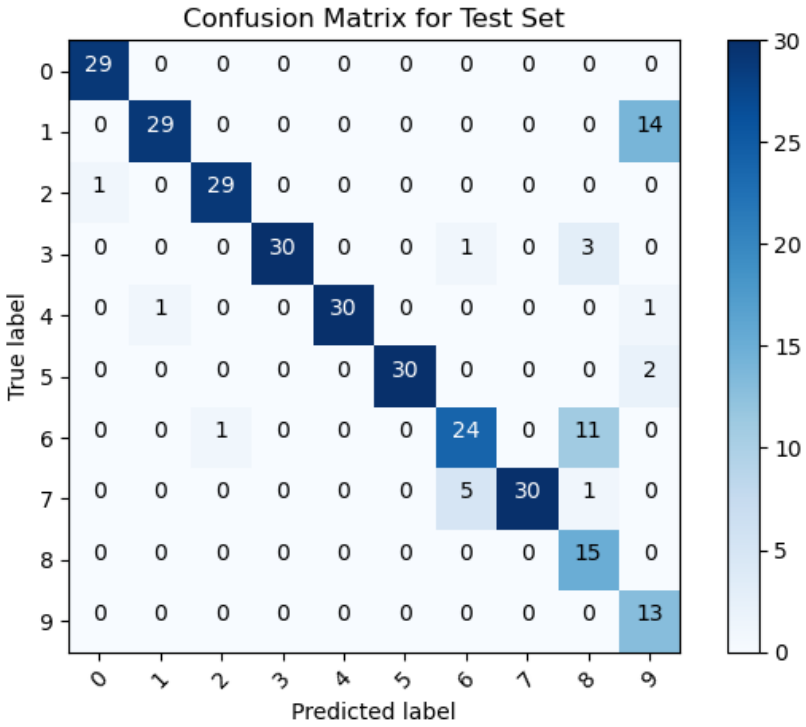
Ερωτήματα 3,4

Για το απλό LSTM ως optimizer επιλέχθηκε ο AdamW με learning rate = 10^{-4}. Ως συνάρτηση σφάλματος επιλέχθηκε η Cross Entropy Loss function καθώς έχουμε να κάνουμε με ένα πρόβλημα ταξινόμησης.

Με το πέρας των 100 επαναλήψεων έχουμε:

validation accuracy	85.74%
test accuracy	86.33%


Training and Validation Losses Over Epochs
Training Loss
Validation Loss
Loss
Epochs
4.00
3.75
3.50
3.25
3.00
2.75
2.50
2.25
0
20
40
60
80
100


Confusion Matrix for Test Set
True label
Predicted label
0 29 0 0 0 0 0 0 0 0 0
1 0 29 0 0 0 0 0 0 0 14
2 1 0 29 0 0 0 0 0 0 0
3 0 0 0 30 0 0 1 0 3 0
4 0 1 0 0 30 0 0 0 0 1
5 0 0 0 0 0 30 0 0 0 2
6 0 0 1 0 0 0 24 0 11 0
7 0 0 0 0 0 0 5 30 1 0
8 0 0 0 0 0 0 0 0 15 0
9 0 0 0 0 0 0 0 0 0 13
0 1 2 3 4 5 6 7 8 9
30
25
20
15
10
5
0

Ερώτημα 5

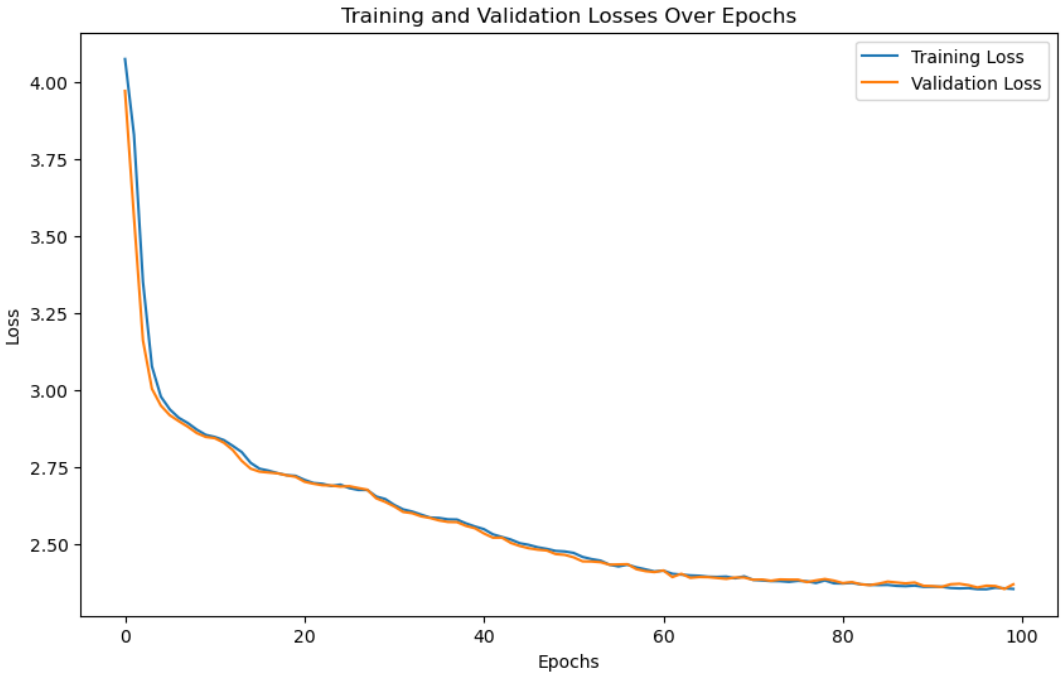
Το Dropout είναι μια τεχνική για regularization η οποία αφαιρεί ένα ποσοστό νευρώνων σε κάθε iteration της εκπαίδευσης με σκοπό να αποτραπεί το φαινόμενο overfitting. Σε ένα LSTM δίκτυο συγκεκριμένα το dropout μπορεί να εφαρμοστεί στις συνδέσεις μεταξύ των κρυφών καταστάσεων ώστε το δίκτυο να μην σχηματίζει συσχετίσεις μεταξύ συγκεκριμένων νευρώνων.

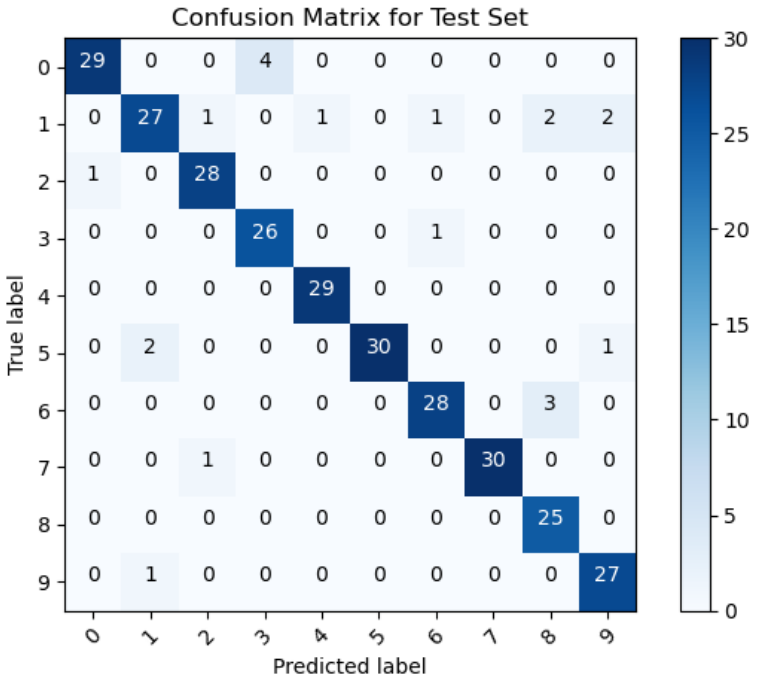
Το L2 Regularization είναι και αυτό μια τεχνική για regularization. Σκοπός της είναι να αποτρέπει τα μεγάλα βάρη από το να επηρεάζουν σε μεγάλο βαθμό το σφάλμα εκπαίδευσης. Κατά τη διάρκεια εκπαίδευσης αποσκοπεί στην ελαχιστοποίηση του σφάλματος αλλά και στη μείωση των βαρών. Αυτό το επιτυγχάνει εισάγοντας μια ποινή (penalty) στο σφάλμα η οποία εξαρτάται από τον συντελεστή λ (weight decay) και το άθροισμα των τετραγώνων των βαρών και παρόλο που αυξάνει το σφάλμα καταλήγει να κάνει το μοντέλο να είναι λιγότερο εξαρτημένο από τα δεδομένα εκπαίδευσης. Ουσιαστικά χρησιμοποιείται και αυτή η τεχνική για αποφυγή overfitting.

Εφαρμόζοντας τις τεχνικές λάβαμε:

validation accuracy	96.11%
test accuracy	93%

Παρατηρούμε ότι έχουμε αρκετά καλύτερα αποτελέσματα στο test accuracy. Επίσης το L2 εφαρμόζει ποινή στα μεγάλα βάρη με αποτέλεσμα να έχουμε μικρότερες αποκλίσεις στη τιμή του σφάλματος κατά την εκπαίδευση, όπως φαίνεται και από το παρακάτω γράφημα.





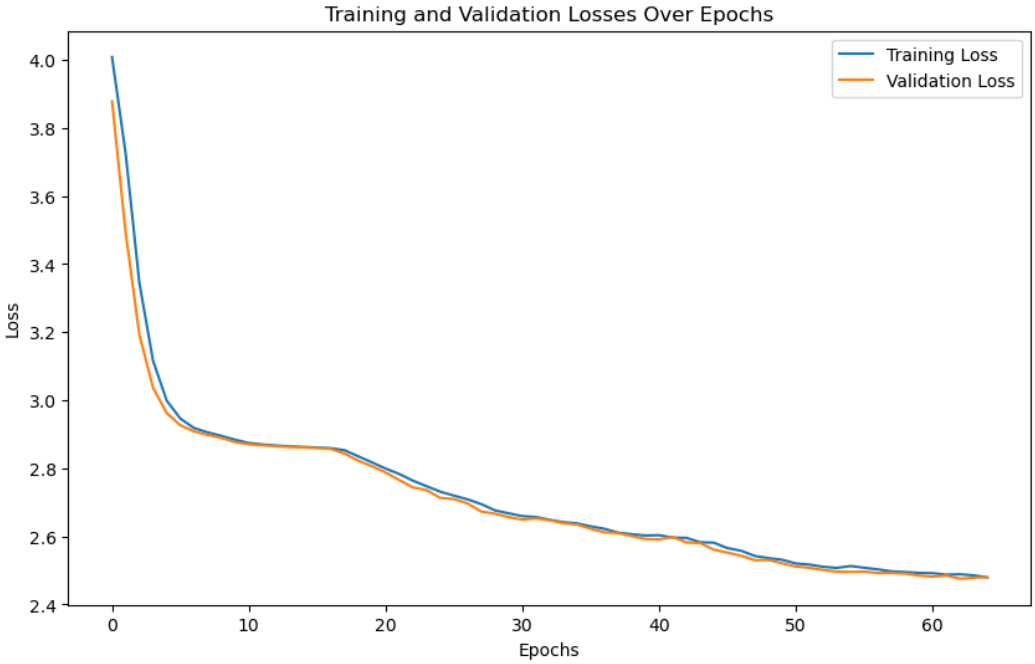
Ερώτημα 6

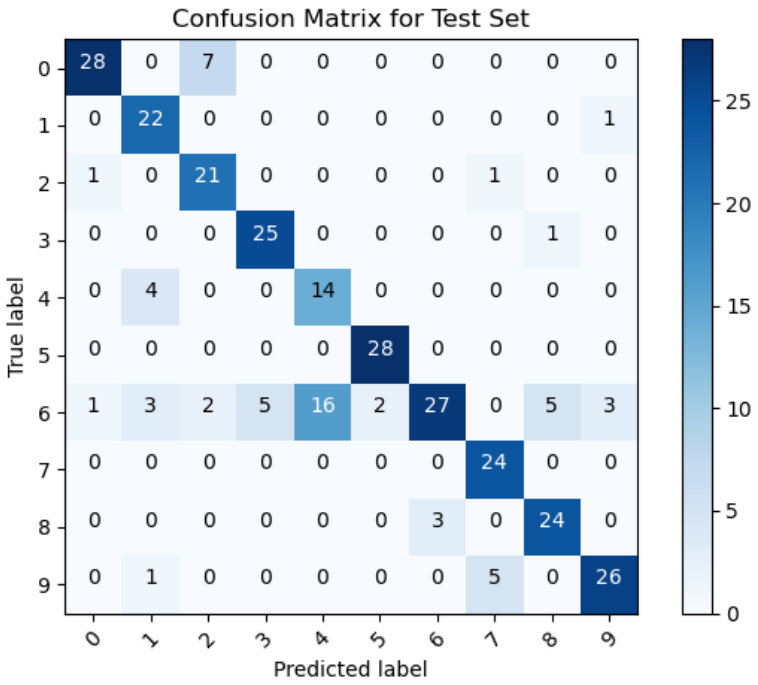
Το Early Stopping είναι μια τεχνική για regularization. Η βασική της λογική προβλέπει τη διακοπή της εκπαίδευσης αν παρατηρηθεί αρνητική βελτίωση για έναν αριθμό συνεχόμενο εποχών (η ανεκτικότητα είναι παράμετρος που ρυθμίζεται). Αποσκοπεί στην αποτροπή overfitting αλλά και στη βελτίωση της απόδοσης του αλγορίθμου καθώς εκτελούνται λιγότερα βήματα κατά τη διάρκεια της εκπαίδευσης.

Χρησιμοποιώντας το με patience = 3 λάβαμε τα εξής:

last epoch	65
validation accuracy	76.11%
test accuracy	79.67%

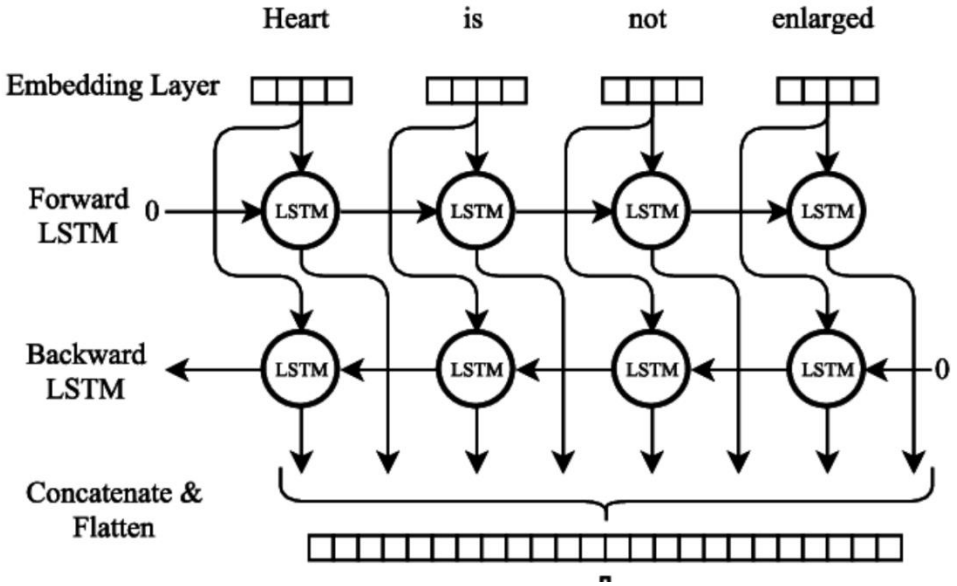
Εκτελέστηκαν 65 εποχές από τις αρχικές 100 (35% μείωση), δηλαδή ο κώδικας είναι αρκετά πιο γρήγορος. Το τίμημα όπως φαίνεται από τον πίνακα ήταν να έχουμε αισθητή πτώση στην ακρίβεια.


Training and Validation Losses Over Epochs
Training Loss
Validation Loss
Loss
Epochs
4.0
3.8
3.6
3.4
3.2
3.0
2.8
2.6
2.4
0
10
20
30
40
50
60


Confusion Matrix for Test Set
True label
Predicted label
0 28 0 7 0 0 0 0 0 0 0
1 0 22 0 0 0 0 0 0 0 1
2 1 0 21 0 0 0 0 1 0 0
3 0 0 0 25 0 0 0 0 1 0
4 0 4 0 0 14 0 0 0 0 0
5 0 0 0 0 0 28 0 0 0 0
6 1 3 2 5 16 2 27 0 5 3
7 0 0 0 0 0 0 0 24 0 0
8 0 0 0 0 0 0 3 0 24 0
9 0 1 0 0 0 0 0 5 0 26
0 1 2 3 4 5 6 7 8 9
25
20
15
10
5
0

Ερώτημα 7

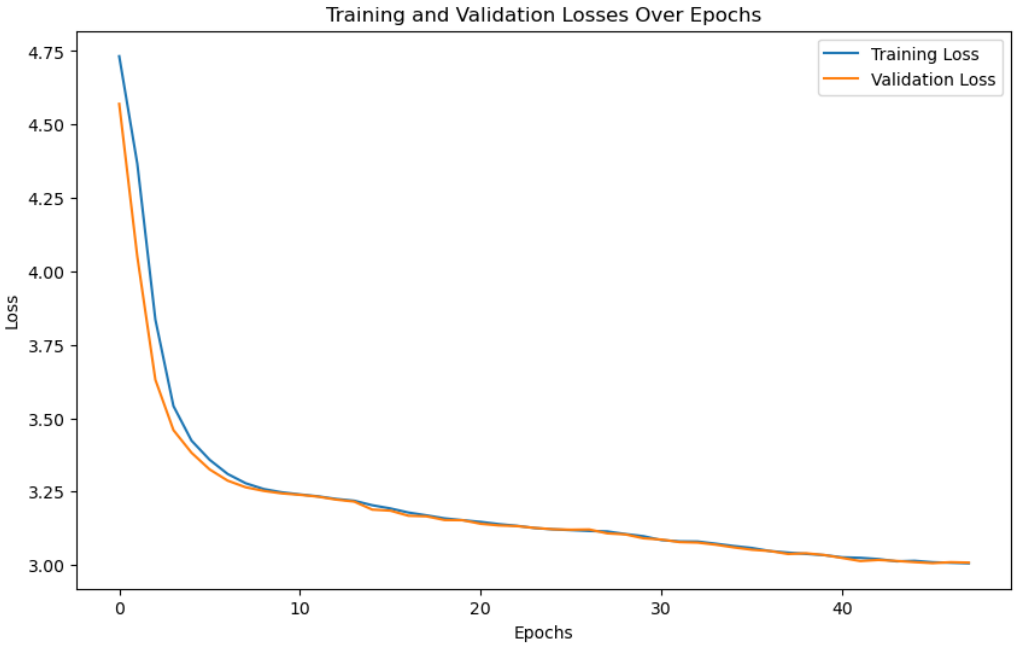
Ένα Bidirectional LSTM περιλαμβάνει ένα επιπλέον layer το οποίο επεξεργάζεται τις ακολουθίες εισόδου στην ανάποδη κατεύθυνση (backward) και τα αποτελέσματα της επεξεργασίας συγχωνεύονται με αυτά της κανονικής (forward) επεξεργασίας. Επιλέγεται όταν έχουμε να κάνουμε με προβλήματα στα οποία η πληροφορία του μέλλοντος είναι εξίσου σημαντική με αυτή του παρελθόντος (όπως η αναγνώριση φωνής. Βασικό μειονέκτημα η αύξηση του χρόνου εκπαίδευσης (ουσιαστικά οι διπλάσιοι υπολογισμοί).

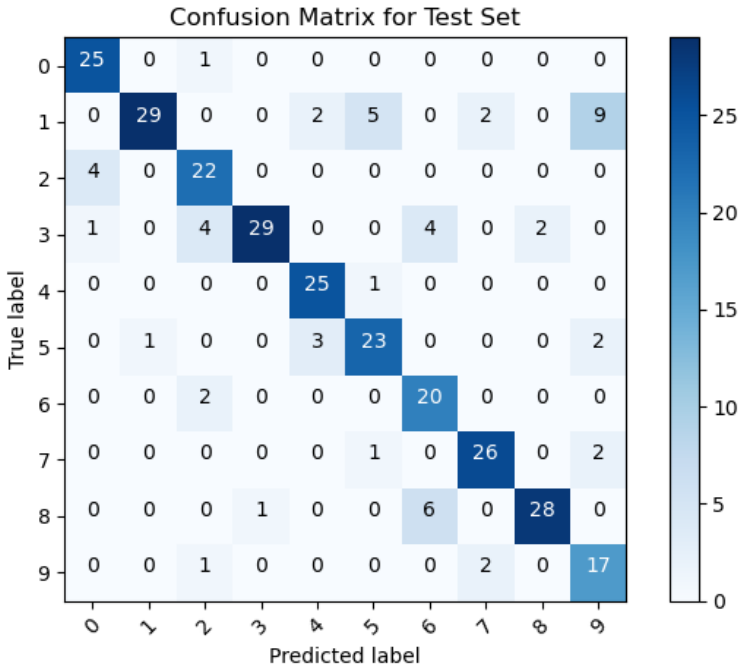


Αξιοποιήσαμε το bidirectional LSTM μαζί με τις προηγούμενες ρυθμίσεις και λάβαμε τα εξής αποτελέσματα:

last epoch	48
validation accuracy	81.48%
test accuracy	81.33%

Παρατηρούμε πως παρόλο που εκτελέστηκαν ακόμα λιγότερες εποχές (48 αντί για 65), η ακρίβεια αυξήθηκε αρκετά. Επομένως το bidirectional μοντέλο αντισταθμίζει την έλλειψη ακρίβειας που εισάγει το Early Stopping.


Training and Validation Losses Over Epochs
Training Loss
Validation Loss
Loss
Epochs
4.75
4.50
4.25
4.00
3.75
3.50
3.25
3.00
0
10
20
30
40


Confusion Matrix for Test Set
True label
Predicted label
0 25 0 1 0 0 0 0 0 0 0
1 0 29 0 0 2 5 0 2 0 9
2 4 0 22 0 0 0 0 0 0 0
3 1 0 4 29 0 0 4 0 2 0
4 0 0 0 0 25 1 0 0 0 0
5 0 1 0 0 3 23 0 0 0 2
6 0 0 2 0 0 0 20 0 0 0
7 0 0 0 0 0 1 0 26 0 2
8 0 0 0 1 0 0 6 0 28 0
9 0 0 1 0 0 0 0 2 0 17
0 1 2 3 4 5 6 7 8 9
25
20
15
10
5
0