Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Αναγνώριση Προτύπων

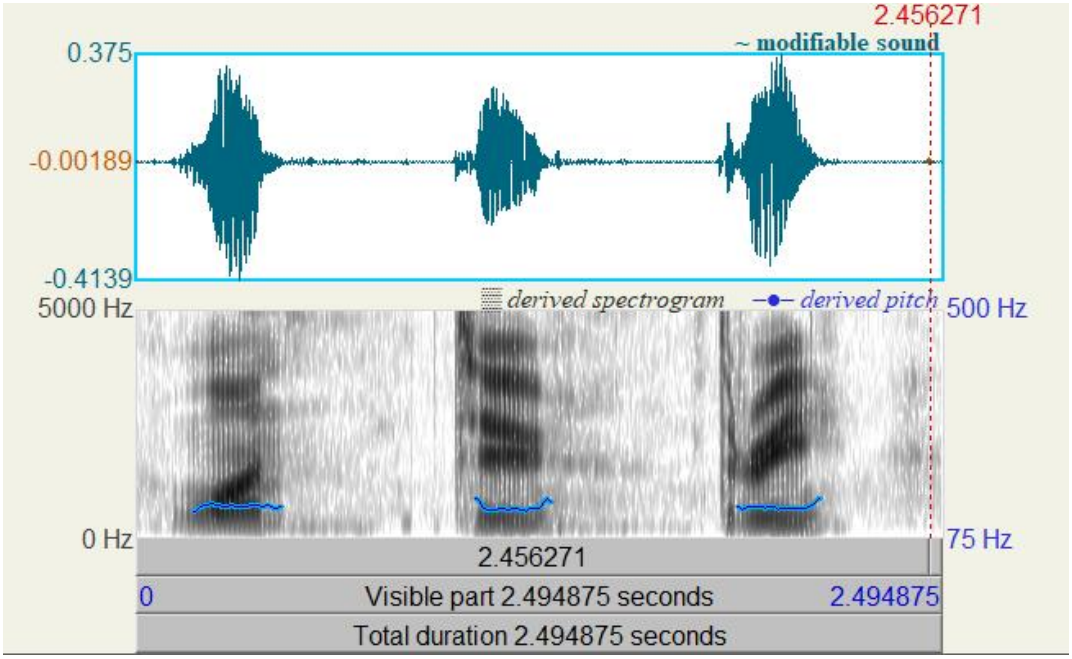
Εξάμηνο 9ο (Εαρινό Εξάμηνο 2022-2023)

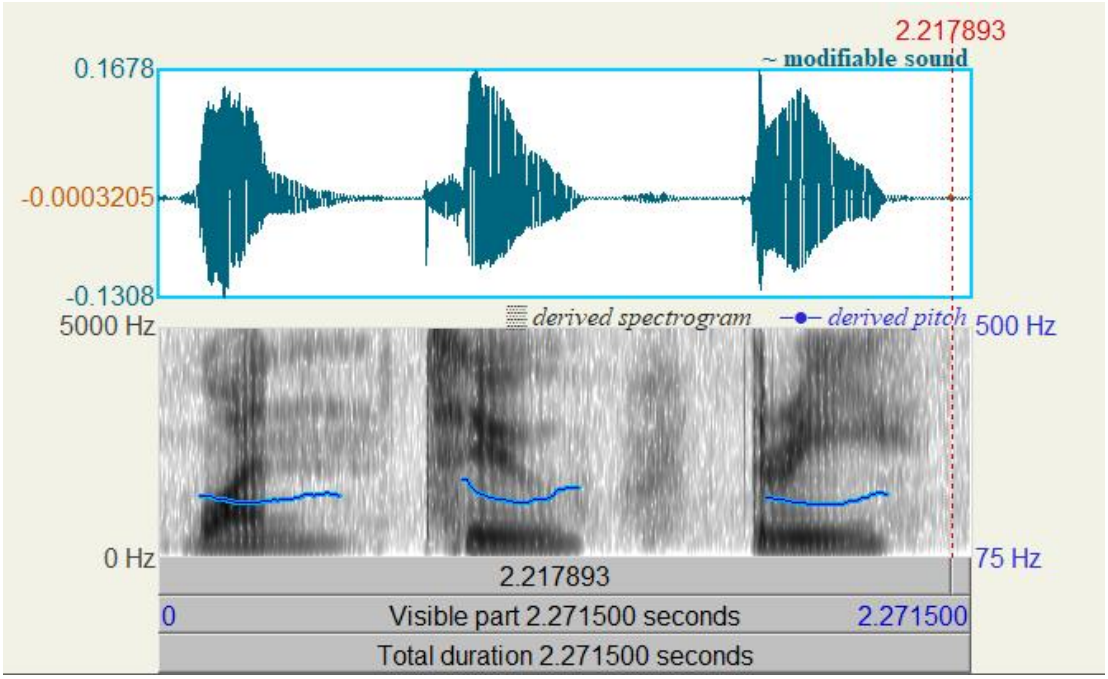
2η Εργαστηριακή Άσκηση

Χάιδος Νικόλαος el18096, Σπανός Νικόλαος el18822

Βήμα 1

Οι κυματομορφές και τα spectrograms των δύο ηχητικών αρχείων, είναι:





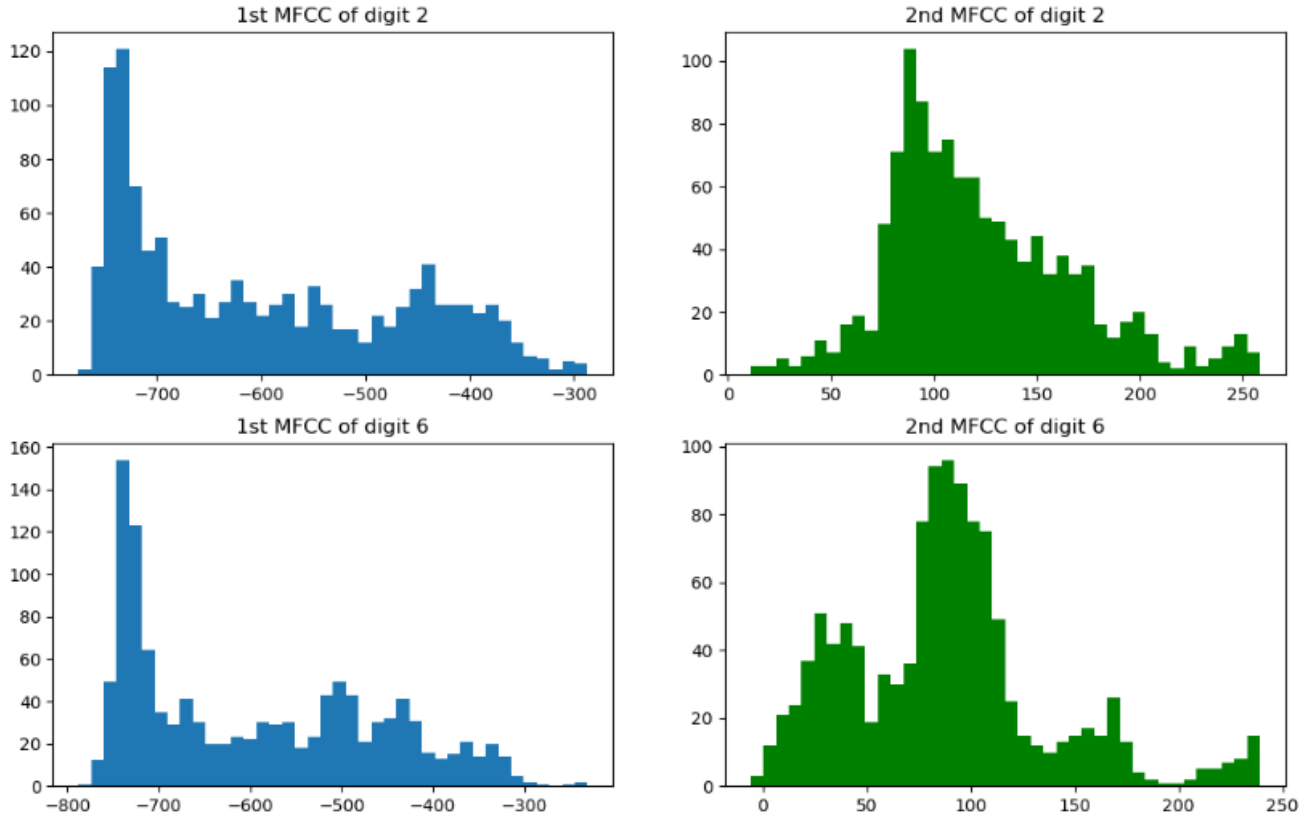
Το πρώτο είναι για τον ομιλητή 1, και το δεύτερο για τον ομιλητή 8. Όσο για τα επιμέρους χαρακτηριστικά του κάθε ηχητικού σήματος:

	Ομιλητής 1 (Hz)	Ομιλητής 8 (Hz)
Μέσο Pitch “α”	134.31	176.25
Μέσο Pitch “ου”	128.82	186.49
Μέσο Pitch “ι”	130.88	178.82
Formants “α”	786.53 1057.93 2338.43	925.39 1664.94 3061.81
Formants “ου”	347.56 1783.35 2353.65	324.43 1670.77 2648.22
Formants “ι”	391.86 2029.38 2517.18	332.52 2182.52 2965.08

Εφόσον, η φωνή του Ομιλητή 8 είναι γυναικεία, αναμένουμε ότι οι συχνότητες θα είναι λίγο πιο υψηλές από αυτές του Ομιλητή 1 (αντρική φωνή), κάτι που βλέπουμε και στα περισσότερα φωνήεντα. Επίσης, λόγω του διαφορετικού μεγέθους/σχήματος των φωνητικών χορδών του άντρα σε σχέση με των γυναικείων φωνητικών χορδών, παρατηρούμε επίσης διαφορές στα formants των τριών φωνήεντων.

Βήμα 4

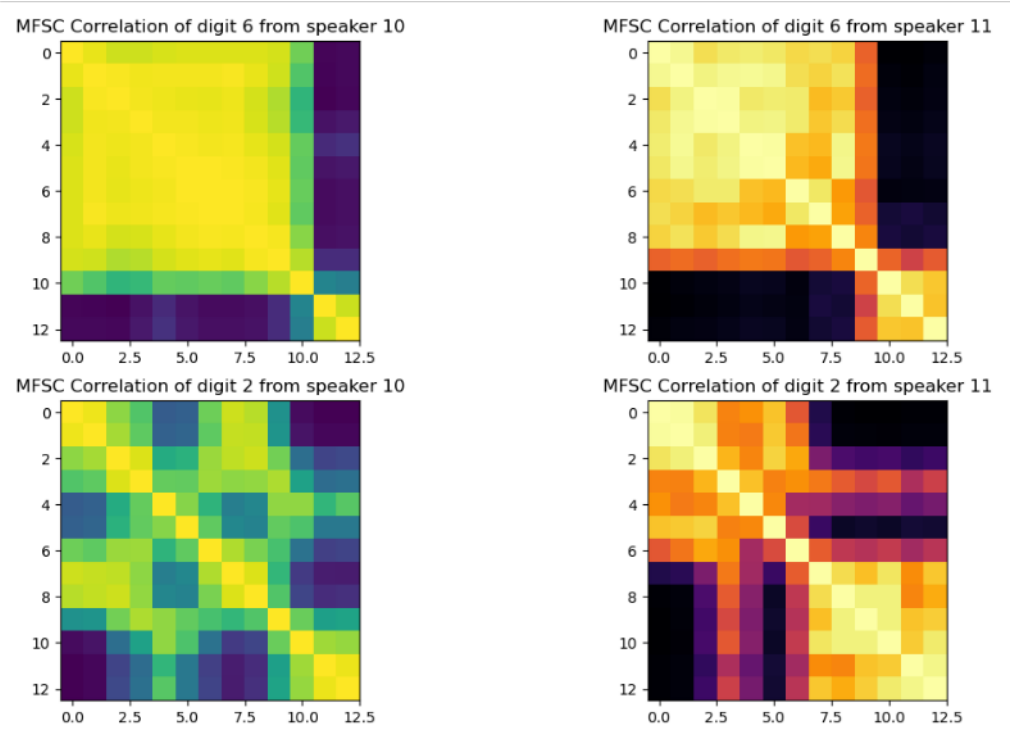
Έχοντας υπολογίσει τα MFCC των ψηφίων, παρουσιάζουμε σε ιστογράμματα τους πρώτους δύο συντελεστές των ψηφίων 2 και 6:



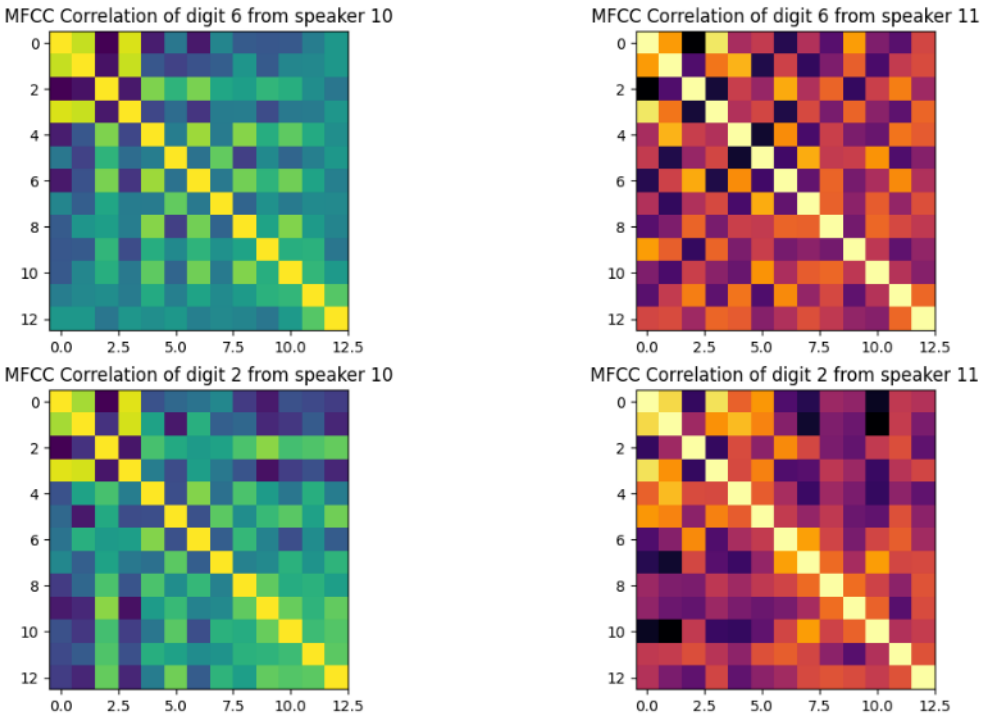
Ανάμεσα στα ίδια MFCC διαφορετικών ψηφίων δεν παρατηρείται μεγάλη απόκλιση, ενώ σε διαφορετικά MFCC είναι αισθητή η διαφορά.

Για τον υπολογισμό των MFSC χρησιμοποιήσαμε την συνάρτηση $melspectrogram$, η οποία εκτελεί την διαδικασία του MFCC χωρίς να εφαρμόζει μετασχηματισμό DCT στο τέλος.

Για τις συσχετίσεις των MFSC παίρνουμε τα εξής αποτελέσματα:



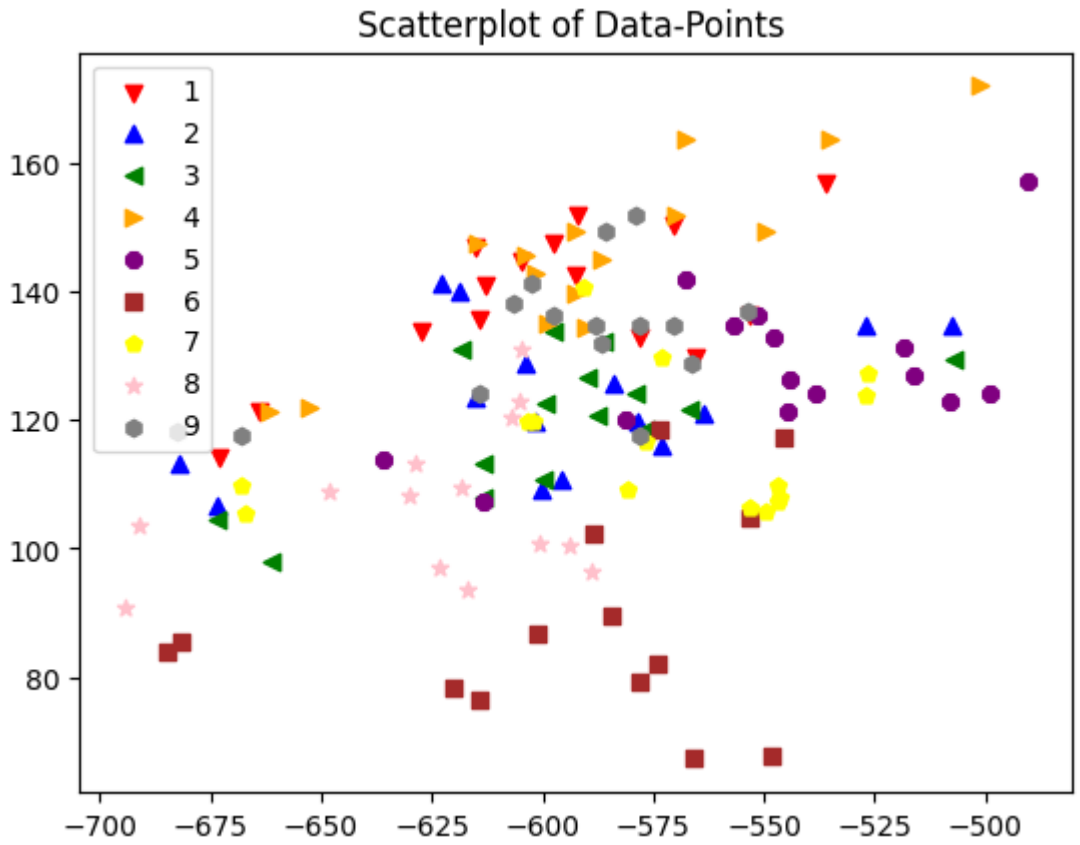
Για τις συσχετίσεις των MFCC παίρνουμε τα εξής αποτελέσματα:



Στα MFSC παρατηρούμε μεγάλη συσχέτιση μεταξύ των συντελεστών, ενώ στο MFCC μικρή συσχέτιση. Έτσι, υπάρχει περισσότερη πληροφορία στα MFCC από ότι στα MFSC και για αυτό είναι προτιμητέα σε διαδικασίες πρόβλεψης.

Βήμα 5

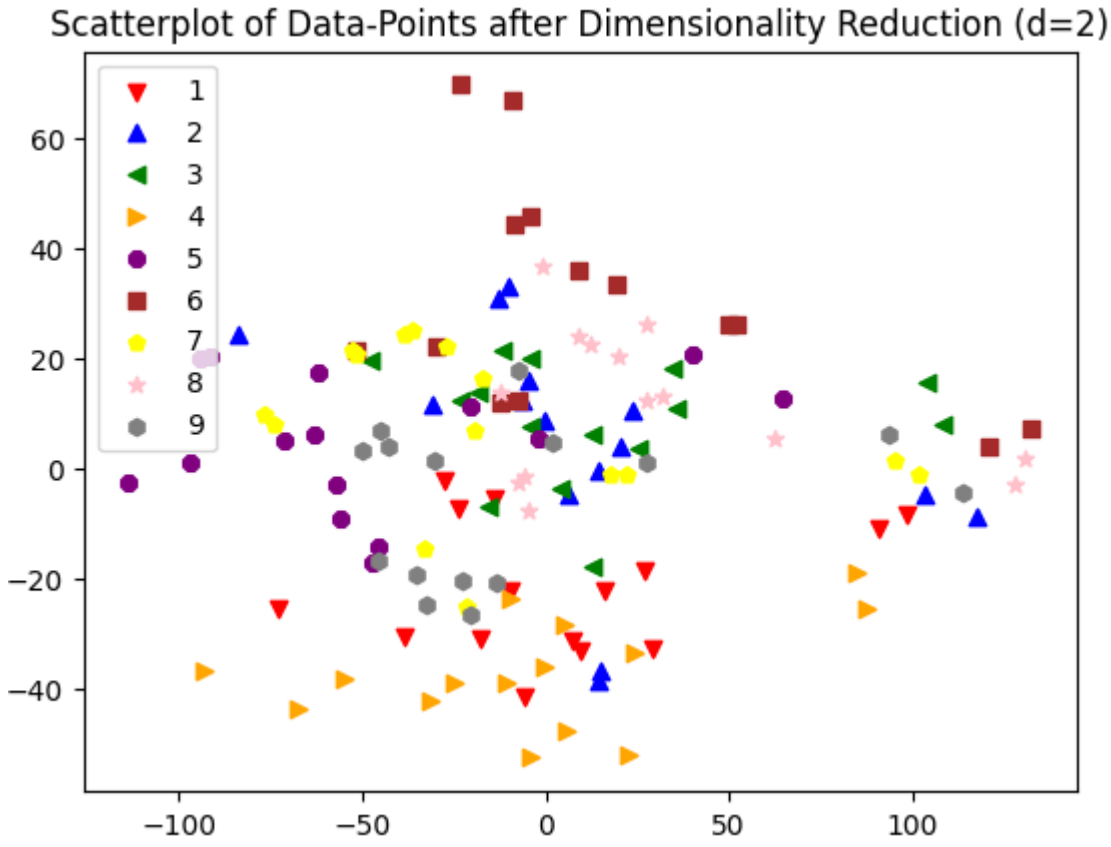
Για την ένωση των MFCC – Delta – Delta-deltas κάναμε concatenate τις μέσες τιμές και τις τυπικές αποκλίσεις του καθενός, υπολογισμένες στον άξονα των παραθύρων. Άρα, συνολικά θα έχουμε $13 * 3 * 2 = 78$ features. Για το scatter-plot, χρησιμοποιώντας τα δύο πρώτα features, πήραμε το εξής αποτέλεσμα:



Παρατηρούμε από το διάγραμμα ότι τα δεδομένα πλησιάζουν στην δημιουργία clusters, αλλά καθώς δεν είναι ξεκάθαρα διαχωρίσιμα η πρόβλεψη γίνεται πιο δύσκολη (ειδικά για γραμμικους ταξινομητές).

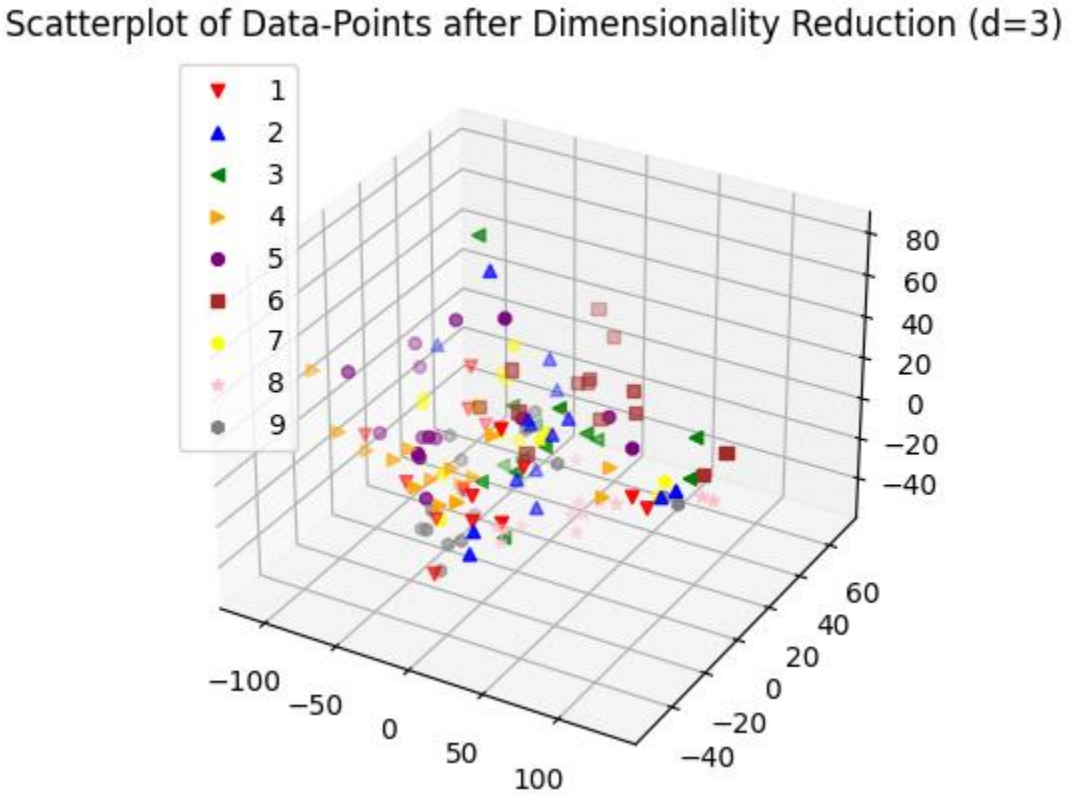
Βήμα 6

Για καλύτερη αναπαράσταση των δεδομένων, εφαρμόσαμε PCA για την μείωση σε δύο διαστάσεις. Το καινούριο scatterplot έχει ως εξής:



Παρατηρούμε ότι τα clusters των ψηφίων είναι πιο ξεκάθαρα διαχωρίσιμα σε σχέση με πριν και στις δύο διαστάσεις διατηρείται το 70.65% της αρχικής διασποράς. Αν και τα δεδομένα βλέπουμε ότι δημιουργούν πιο ξεκάθαρα διαχωρισμένα clusters, χάνεται ένα αρκετά σημαντικό ποσοστό της αρχικής πληροφορίας.

Επαναλαμβάνουμε την ίδια διαδικασία για 3 διαστάσεις και παίρνουμε το εξής αποτέλεσμα:



Παρομοίως, τα clusters είναι πιο ξεκάθαρα και παράλληλα περιέχεται παραπάνω πληροφορία, καθώς στις 3 διαστάσεις διατηρείται το 81.49% της αρχικής διασποράς.

Συνολικά, η μείωση της διαστατικότητας βοηθάει στην καλύτερη αναπαράσταση των δεδομένων, αλλά με tradeoff στην συνολική πληροφορία που χάνεται ($20\% - 30\%$ απώλεια).

Βήμα 7

Για την κανονικοποίηση των δεδομένων χρησιμοποιήθηκε *ο StandardScaler*. Χρησιμοποιώντας το αρχικό dataset (για να διατηρήσουμε όλη την πληροφορία που παίρνουμε από τα MFCCs), εκπαιδεύσαμε 5 διαφορετικούς ταξινομητές και πήραμε τα παρακάτω αποτελέσματα:

Classifiers	Score
Custom NaiveBayes	65.00%
SKLearn NaiveBayes	65.00%
SKLearn SVM(rbf)	57.50%
SKLearn MLP	50.00%
SKLearn RandomForest	80.00%

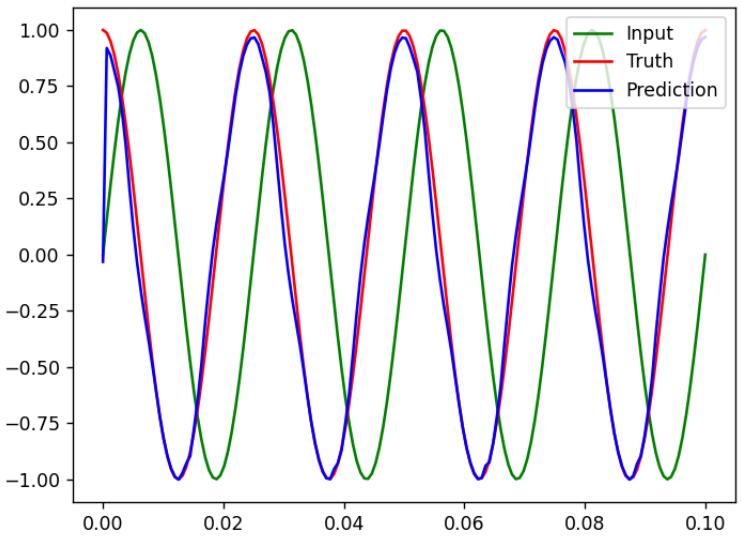
Βέβαια, λόγω του μικρού πλήθους δειγμάτων, βλέπουμε ότι γενικά τα score δεν ήταν υψηλά, και η εκπαίδευση των ταξινομητών ήταν αρκετά εξαρτώμενη από το τυχαίο split στο dataset.

Βήμα 8

Υλοποιήσαμε την κλάση *CustomRNN* που υποστηρίζει τρία είδη RNN (Vanilla RNN, LSTM και GRU). Παρακάτω θα παρουσιάσουμε τα αποτελέσματα της πρόβλεψης για κάθε RNN. Το training έγινε σε 200 εποχές και για τα τρία RNN, με 2 stacked RNNs και hidden size ίσο με 20.

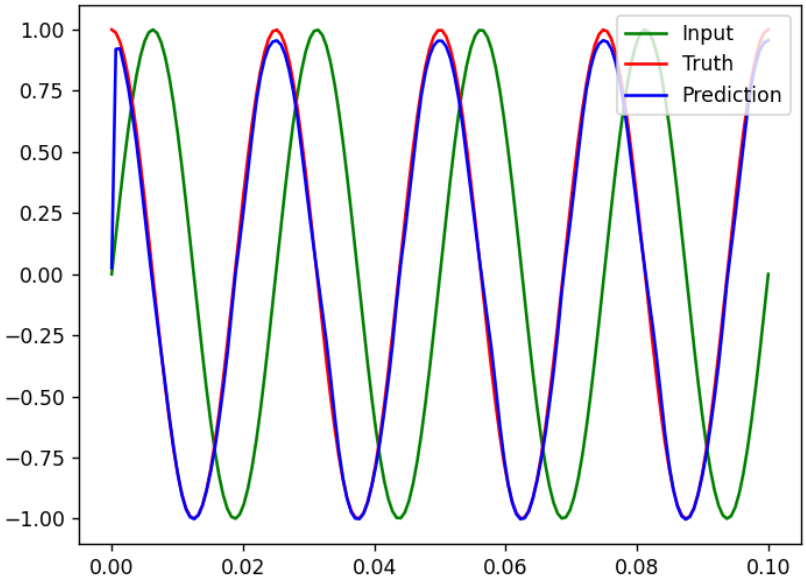
Vanilla RNN:

Το τελικό loss ήταν ίσο με 0.053 και το τελικό διάγραμμα πρόβλεψης είναι το εξής:



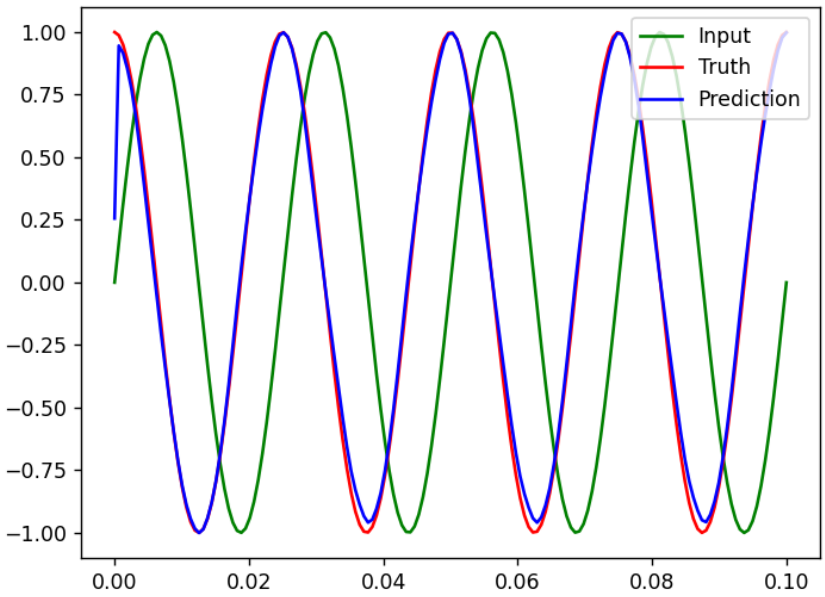
LSTM:

Το τελικό loss ήταν ίσο με 0.049 και το τελικό διάγραμμα πρόβλεψης είναι το εξής:



GRU:

Το τελικό loss ήταν ίσο με 0.049 και το τελικό διάγραμμα πρόβλεψης είναι το εξής:



Γενικότερα και οι τρεις υλοποιήσεις έχουν παρόμοια αποτελέσματα, όμως το LSTM και το GRU παρουσιάζουν πιο ομοιόμορφες κυματομορφές σε σχέση με το Vanilla RNN. Ο λόγος που οι δύο τελευταίες υλοποιήσεις είναι πιο διαδεδομένες είναι διότι δημιουργήθηκαν για να λύσουν δύο μεγάλα προβλήματα του Vanilla RNN, το vanishing και το exploding gradient.

Βήμα 9

Για το $stratified\,split$, χρησιμοποιούμε την παράμετρο $stratify$ της συνάρτησης $train\_test\_split$, η οποία μάς εγγυάται ότι θα έχουμε ίδια κατανομή των Labels σε Train-Set και Test-Set.

Βήμα 10

Για την αρχικοποίηση του Πίνακα Μεταβάσεων a_{ij}, ορίσαμε τις τιμές ως εξής:

- $a_{ii} = a_{i,i+1} = 0.5 \quad \gamma\iota\alpha\ i < n_{states}$ (η πιθανότητα αθροίζει σε μονάδα)
- $a_{ii} = 1.0 \quad \gamma\iota\alpha\ i = n_{states}$ (η τελική κατάσταση δεν θα μεταβεί κάπου αλλού)

Για την αρχικοποίηση των Αρχικών Πιθανοτήτων των καταστάσεων π_i, ορίσαμε τις τιμές ως εξής:

- $\pi_i = 0 \quad \gamma\iota\alpha\ i > 0$
- $\pi_i = 1 \quad \gamma\iota\alpha\ i = 0$ (θα ξεκινάει πάντα από την πρώτη κατάσταση)

Για την αρχικοποίηση των Τελικών Πιθανοτήτων των καταστάσεων e_i, ορίσαμε τις τιμές ως εξής:

- $e_i = 0 \quad \gamma\iota\alpha\ i < n_{states}$
- $e_i = 1 \quad \gamma\iota\alpha\ i = n_{states}$ (θα τελειώνει πάντα στην τελευταία κατάσταση)

Βήμα 12

Η διαδικασία του *HyperParameter Optimization* με χρήση του Validation Set, γίνεται για να μπορούμε να κάνουμε τις βέλτιστες επιλογές στην σχεδίαση του μοντέλου μας, έτσι ώστε να πετυχαίνουμε την μέγιστη απόδοση. Ο λόγος που χρησιμοποιούμε το Validation Set, και όχι το Test Set, είναι για να μην κάνουμε tuning του μοντέλου πάνω στο Test Set, διότι αυτό θα

οδηγούσε σε data leakage από το Test Set στην διαδικασία της εκπαίδευσης. Αυτό είναι απαγορευτικό, αφού το Test Set χρησιμοποιείται μόνο για την εκτίμηση της απόδοσης του μοντέλου, και όχι την εκπαίδευσή του, ή την σχεδίασή του.

Με χρήση του Validation Set, κάναμε Grid-Search στις παραμέτρους, ως εξής:

	Min	Max	Step
Number of GMM Mixtures	1	5	1
Number of HMM States	1	4	1
Max Iterations for EM	2	63	3

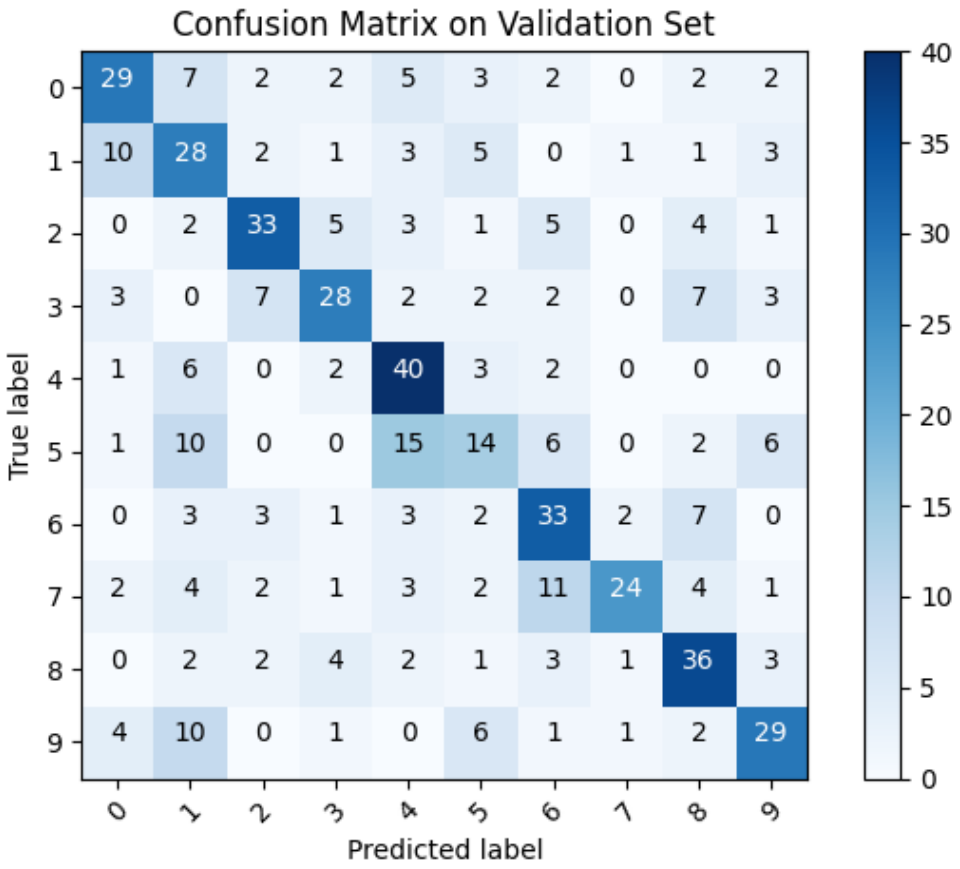
Εν τέλει, το καλύτερο μοντέλο είχε τις εξής τιμές των υπερπαραμέτρων:

- *3 GMM Mixtures*
- *4 HMM States*
- *50 Max Iterations*

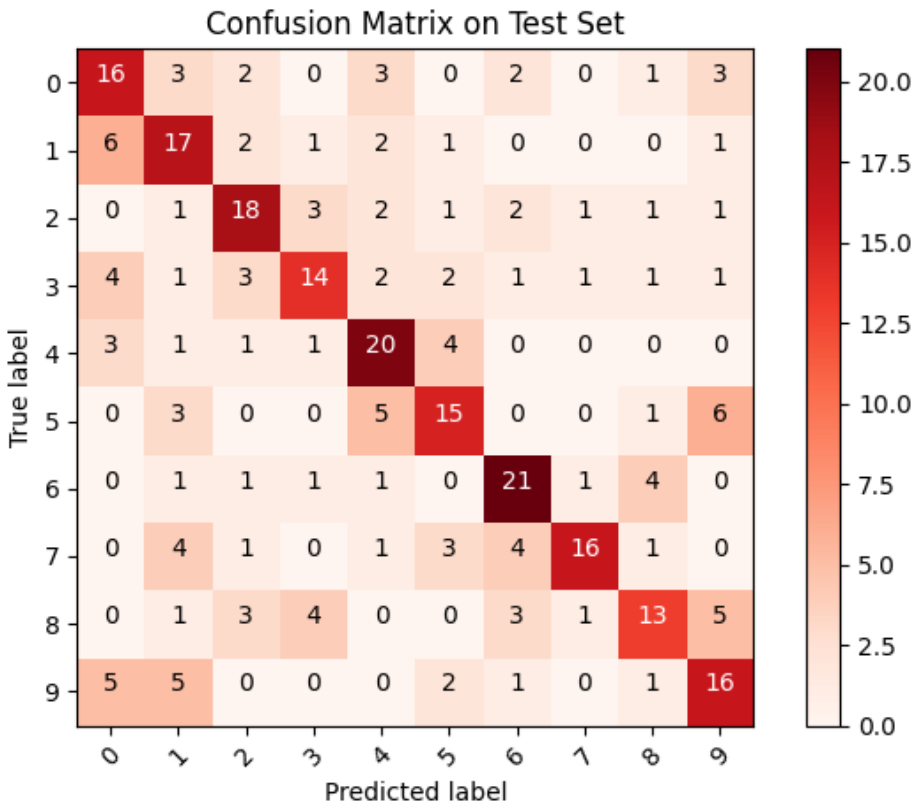
Και πέτυχε Accuracy 54.4% στο Validation Set.

Βήμα 13

Το παραπάνω μοντέλο, πέτυχε 54.44% συνολικό Accuracy στο Validation Set, και ο Confusion Matrix είναι ο εξής:



Το ίδιο μοντέλο, πέτυχε 55.33% συνολικό Accuracy στο Test Set, και ο Confusion Matrix είναι ο εξής:



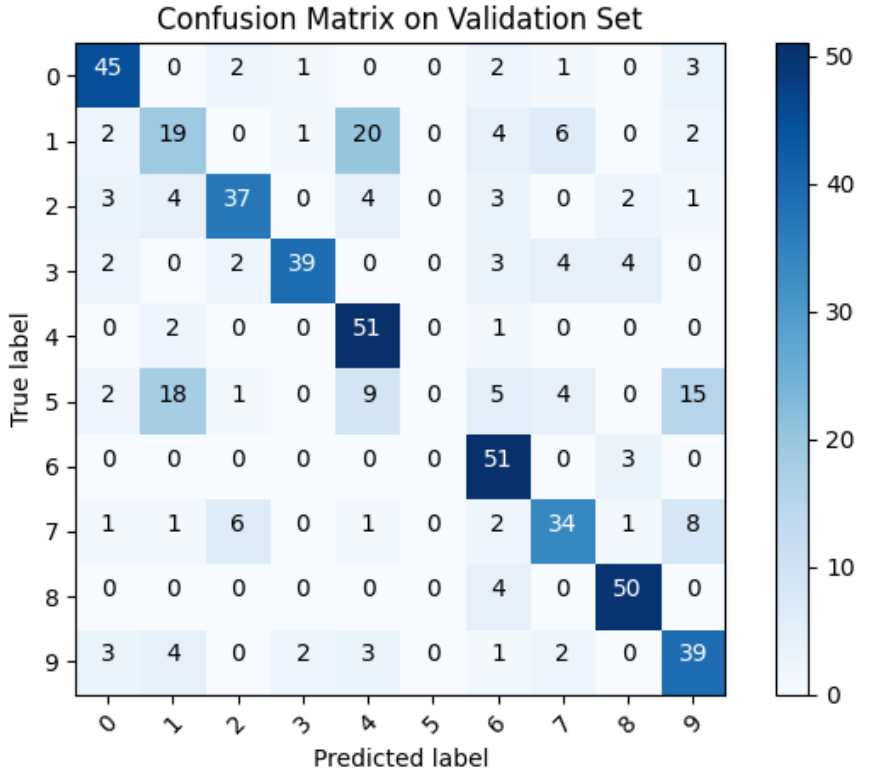
Βήμα 14

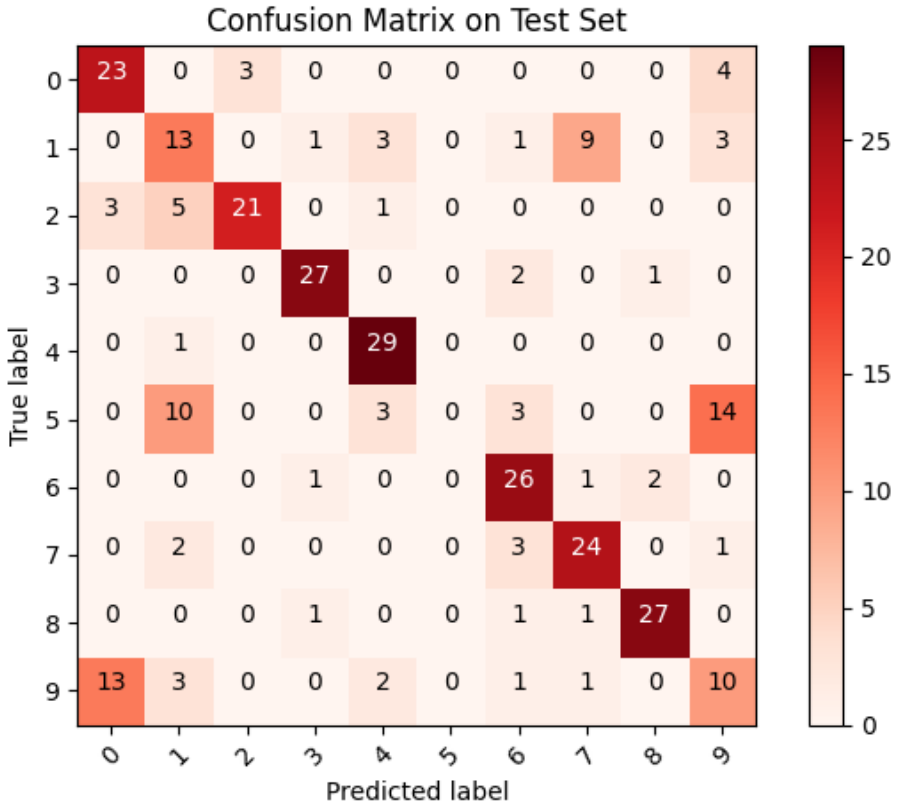
Για την υλοποίηση του *CustomLSTM* μοντέλου, χρησιμοποιήσαμε 2 layers LSTM, 2 layers Dropout, 2 Fully Connected Layers και ενα layer Softmax για την έξοδο. Για το training του νευρωνικού, χρησιμοποιήσαμε Optimizer AdamW με Exponential_LR scheduler, καθώς παρατηρήθηκε από τις δοκιμές ότι είχαν καλύτερη επίδοση πάνω στα δεδομένα. Παράλληλα, το μοντέλο εκπαιδεύτηκε για 100 εποχές και χρησιμοποιήσαμε το pack_padded_sequence για γίνει ταχύτερη η διαδικασία εκπαίδευσης του μοντέλου. Για τις 4 διαφορετικές βελτιώσεις του μοντέλου, πήραμε τα παρακάτω αποτελέσματα:

4. Απλό LSTM

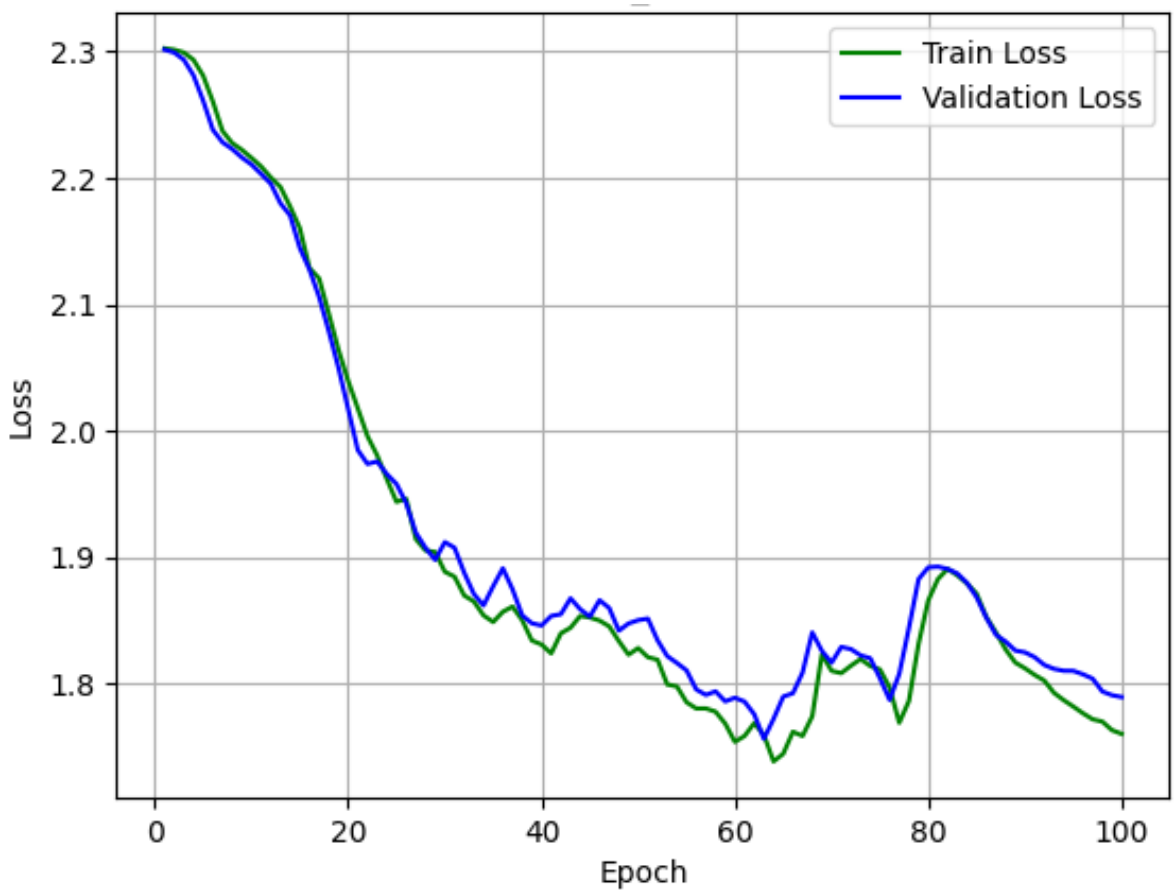
Validation Set Accuracy Μοντέλου: 67.59%
Test Set Accuracy Μοντέλου: 66.66%

Confusion Matrices:





Γράφημα Training και Validation Loss ως προς τις εποχές:

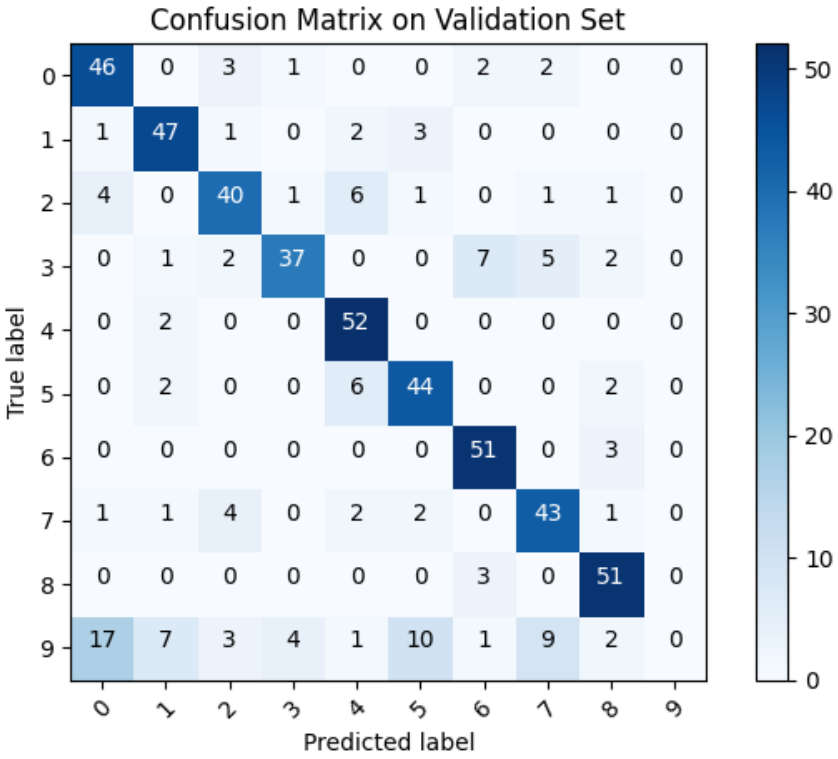


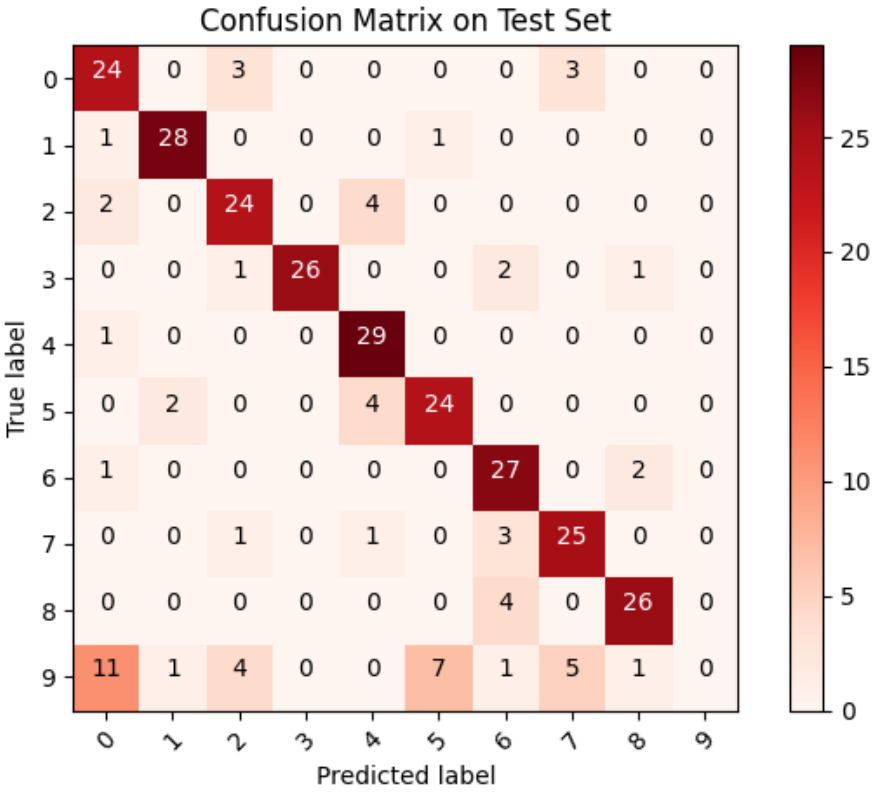
5. **Πρόσθεση Dropout και L2 Regularization**

   Το Dropout Layer μηδενίζει με μια πιθανότητα τον κάθε νευρώνα εισόδου του layer. Αυτό βοηθάει στην επίτευξη καλύτερης γενίκευσης του μοντέλου, εφόσον δεν επικεντρώνεται μόνο σε συγκεκριμένα χαρακτηριστικά, αλλά εκμεταλλεύεται όλη την είσοδο. Το L2 Regularization προσθέτει έναν παραπάνω όρο στο LossFunction, ο οποίος είναι ίσος με την L2 νόρμα των βαρών του δικτύου, ώστε τα βάρη να μην αποκτούν μεγάλες τιμές. Αυτό χρησιμοποιείται, επίσης, για την καλύτερη γενίκευση του μοντέλου. Για το συγκεκριμένο μοντέλο, χρησιμοποιήσαμε πιθανότητα 30% για το Dropout Layer και συντελεστή 10^{-3} για το L2 Regularization.

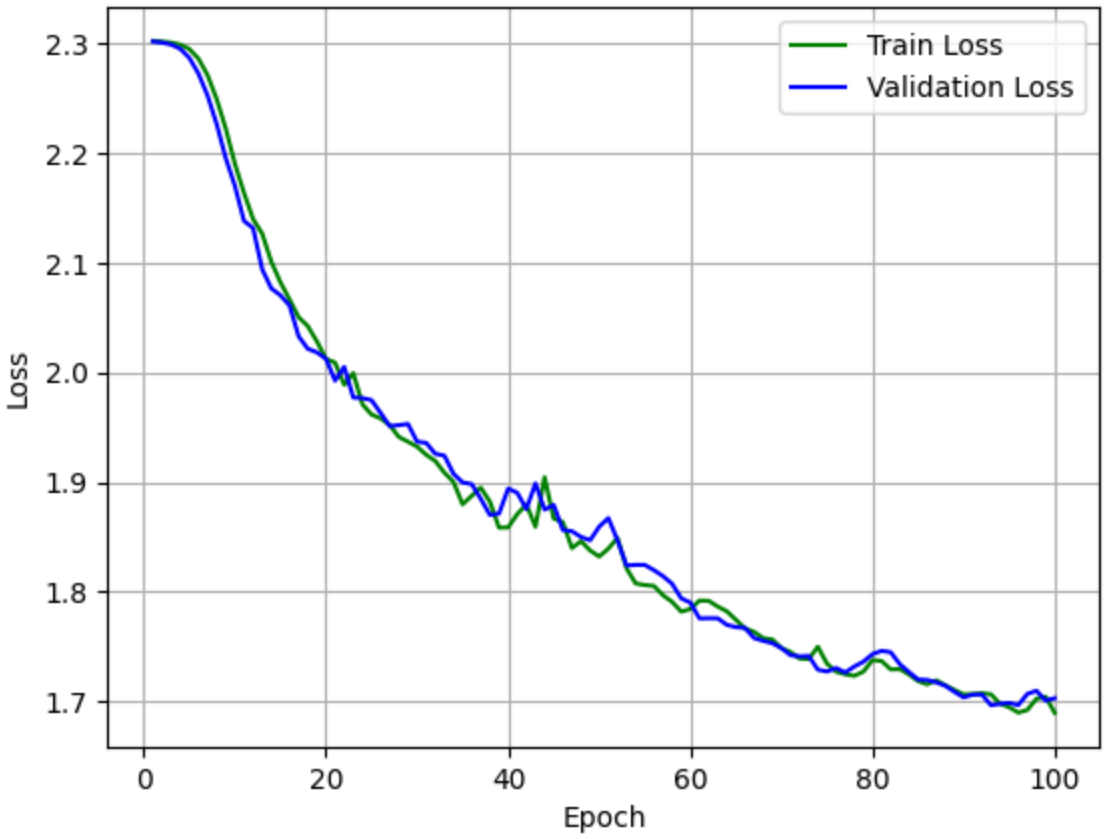
   Validation Set Accuracy Μοντέλου: 76.11%
   Test Set Accuracy Μοντέλου: 77.66%

Confusion Matrices:





Γράφημα Training και Validation Loss ως προς τις εποχές:

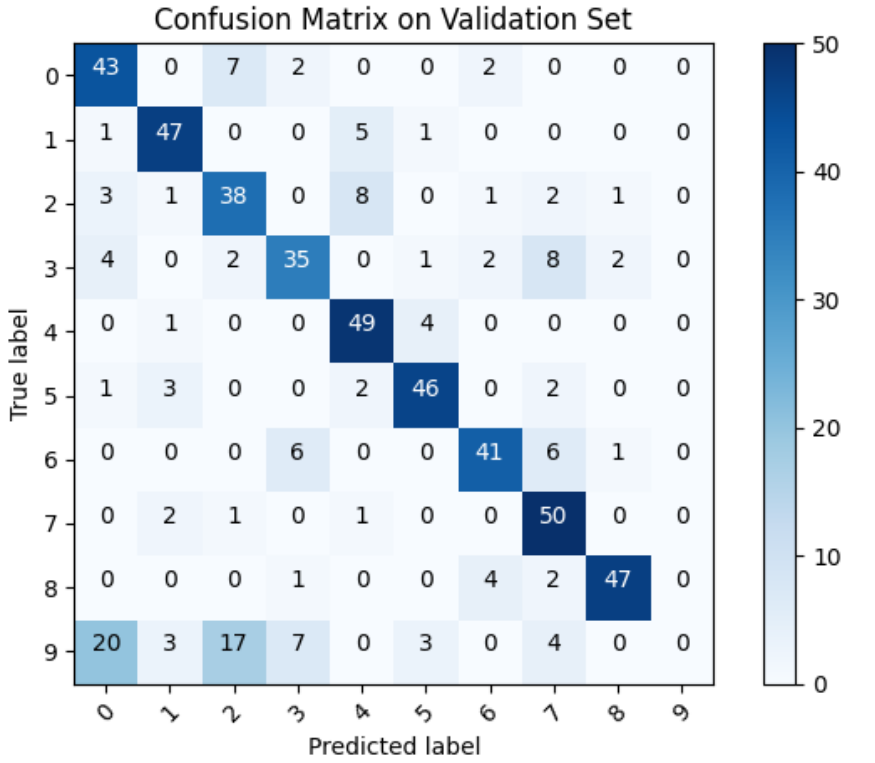


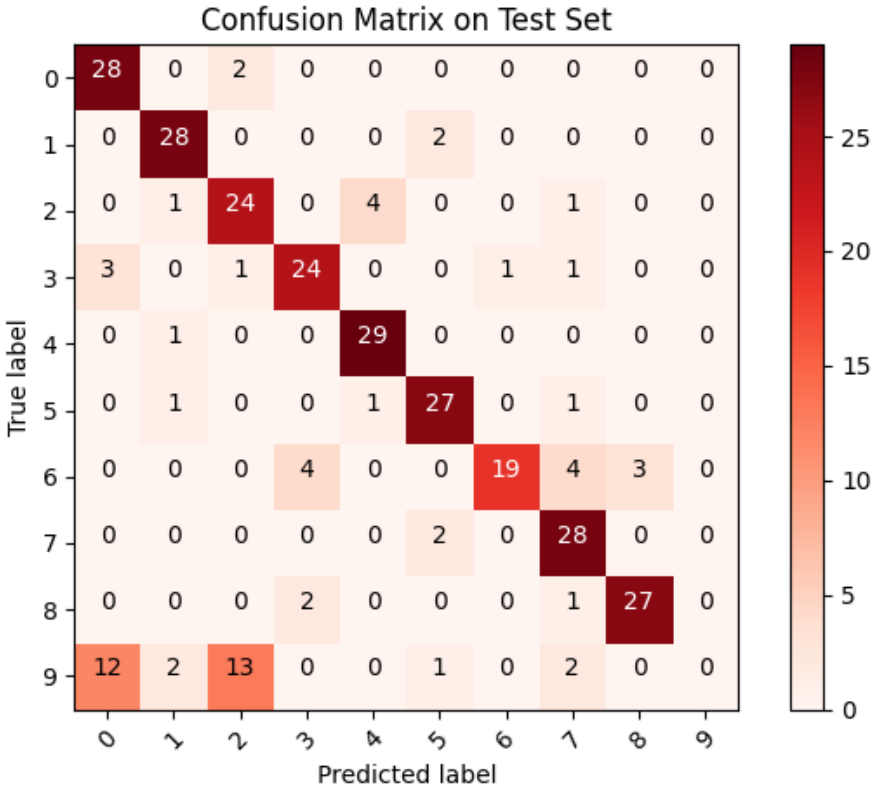
6. **Πρόσθεση Early Stopping και Checkpoints**

   Το EarlyStopping χρησιμοποιείται έτσι ώστε σε περίπτωση που το validation loss ξεκινήσει να αυξάνεται για πλήθος εποχών που υπερβαίνουν ένα όριο, να σταματήσει η εκπαίδευση του μοντέλου και να αποθηκευτούν τα βέλτιστα βάρη. Αυτό βοηθάει στο να αποφύγουμε περιπτώσεις overfitting και το μοντέλο να πετύχει καλύτερο generalization. Η εκπαίδευση του συγκεκριμένου μοντέλου σταμάτησε στην εποχή 76, έχοντας θέσει το όριο εποχών ίσο με 7.

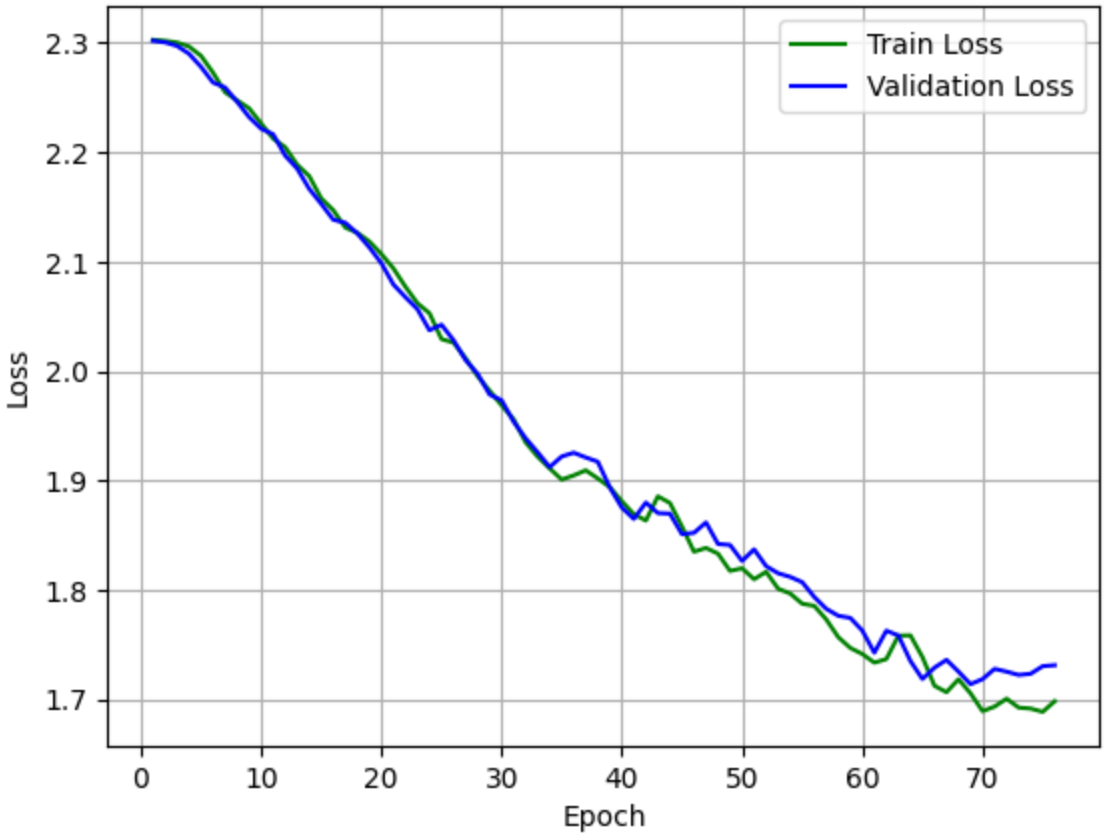
   Validation Set Accuracy Μοντέλου: 73.3%
   Test Set Accuracy Μοντέλου: 78%

Confusion Matrices:





Γράφημα Training και Validation Loss ως προς τις εποχές:

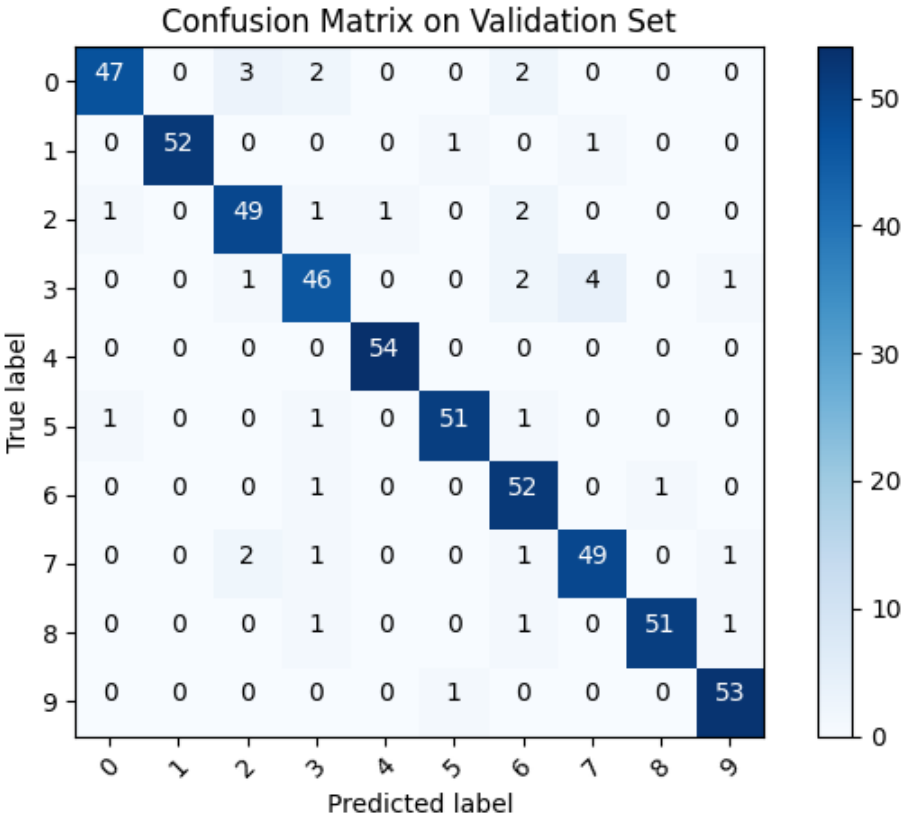


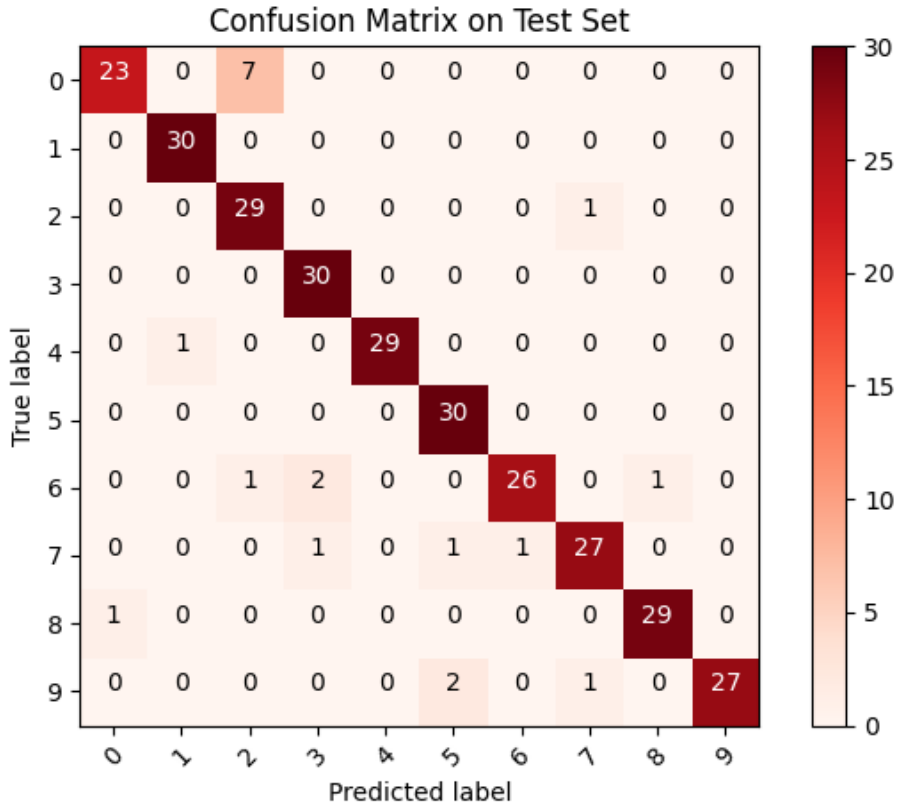
7. **Πρόσθεση Bidirectional**

   Με χρήση μεθόδου Bidirectional, κάθε δείγμα τροφοδοτείται στο LSTM με δύο περάσματα, ένα από την αρχή προς το τέλος και ένα από το τέλος προς την αρχή. Αυτή η μέθοδος κάνει πιο δυνατό το μοντέλο, καθώς μπορεί να συσχετίσει τις μετρήσεις ενός δείγματος και από τις δύο κατευθύνσεις.

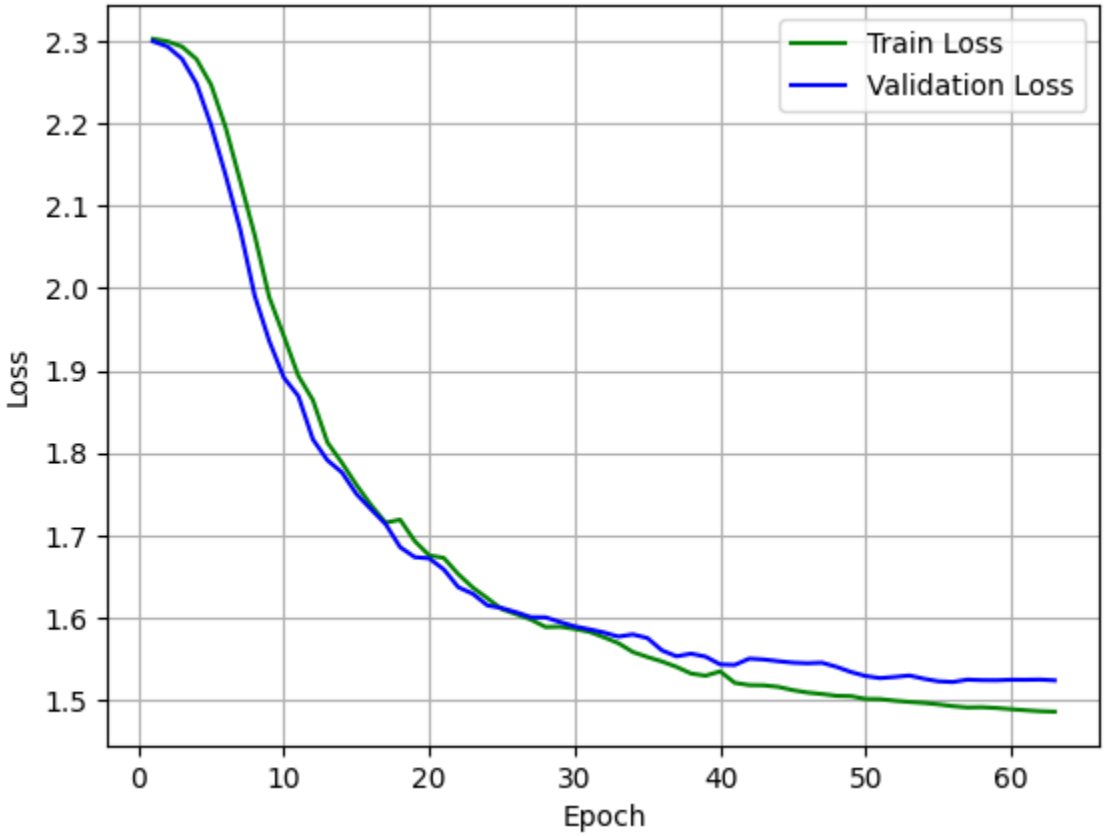
   Validation Set Accuracy Μοντέλου: 93.3%
   Test Set Accuracy Μοντέλου: 93.3%

Confusion Matrices:





Γράφημα Training και Validation Loss ως προς τις εποχές:



Η καλύτερη βελτίωση πάνω στο δίκτυο ήταν με χρήση της μεθόδου Bidirectional, με την οποία παρατηρήσαμε αύξηση του accuracy κατά 15%.