Εργασία

ΜΑΘΗΜΑ: Επεξεργασία Σημάτων Φωνής και Ήχου

Όνομα: ***Μουστάκας Αναστάσιος***

Μητρώο: ***Π16194***

ΠΕΡΙΕΧΟΜΕΝΑ

Δεδομένα

Τα δεδομένα που πήραμε βρίσκονται στο link https://github.com/Jakobovski/free-spoken-digit-dataset και περιέχουν 3000 ηχογραφήσεις από 6 διαφορετικούς ανθρώπους με κάθε νούμερο να υπάρχει 50 φορές οπότε 50*6*10 = 3000 σήματα.

Τα δεδομένα αποθηκεύονται στον φάκελο “Files” ο οποίος πρέπει να είναι στον ίδιο χώρο με το αρχείο ergasia.py που περιέχει την λύση.

Προεπεξεργασία

Πρώτο βήμα είναι το να φορτώσουμε τα δεδομένα. Ακολούθως θα πρέπει να γίνει μια δειγματοληψία για να ρίξουμε τόσο το ρυθμό του σήματος όσο και τα δείγματα καθώς θα βοηθήσει στην επίλυση του αλγορίθμου.

Η συχνότητα που επιλέξαμε να κάνουμε δειγματοληψία είναι τα 8000Hz καθώς είναι αρκετά για να αναπαραστήσουν το σήμα μας σωστά.

Επίσης θα υπολογίσουμε την θεμελιώδη συνάρτηση για κάθε σήμα μέσω της συνάρτησης αυτοσυσχέτισης που θα υπολογίσουμε και θα εμφανίσουμε για κάποια σήματα την τιμή της.

Χρησιμοποιούμε python και τις βιβλιοθήκες **scipy numpy και matplotlib**

Χρησιμοποιούμε την βιβλιοθήκη scipy για να φορτώσουμε τα σήματα με την συνάρτηση load όπου καταλήγουν στην μεταβλητή data.

Έπειτα δημιουργούμε και την μεταβλητή Y που περιέχει την κλάση, δηλαδή το ψηφίο της κάθε ηχογράφησης.

Η συνάρτηση resample κάνει την δειγματοληψία στα 8000Hz και το συνολικό σήμα γίνεται append στην data, αντικαθιστώντας το αρχικό σήμα, για κάθε δείγμα.

Επίσης τώρα υπολογίζουμε την θεμελιώδη συχνότητα ως:

Correlation του σήματος με τον εαυτό του με την συνάρτηση correlate.

Τα μέγιστα υπολογίζονται με την συνάρτηση find_peaks

Η συχνότητα f0 είναι ίση με τη συχνότητα δειγματοληψίας/το μέγιστο της find_peas

Για 4 διαφορετικά δείγματα εμφανίζουμε τα αποτελέσματα

```
Fundamental frequency of signal 100: 1333.3333333333333
Fundamental frequency of signal 300: 8000.0
Fundamental frequency of signal 400: 1000.0
Fundamental frequency of signal 600: 1333.3333333333333
```

Γίνεται και μια τελευταία ενέργεια, βάζοντας 0 σε όλα τα σήματα στο τέλος τους έτσι ώστε να έχουν όλα ίδιο μέγεθος, όπως και το μεγαλύτερο σε μήκος σήμα.

Spectograms

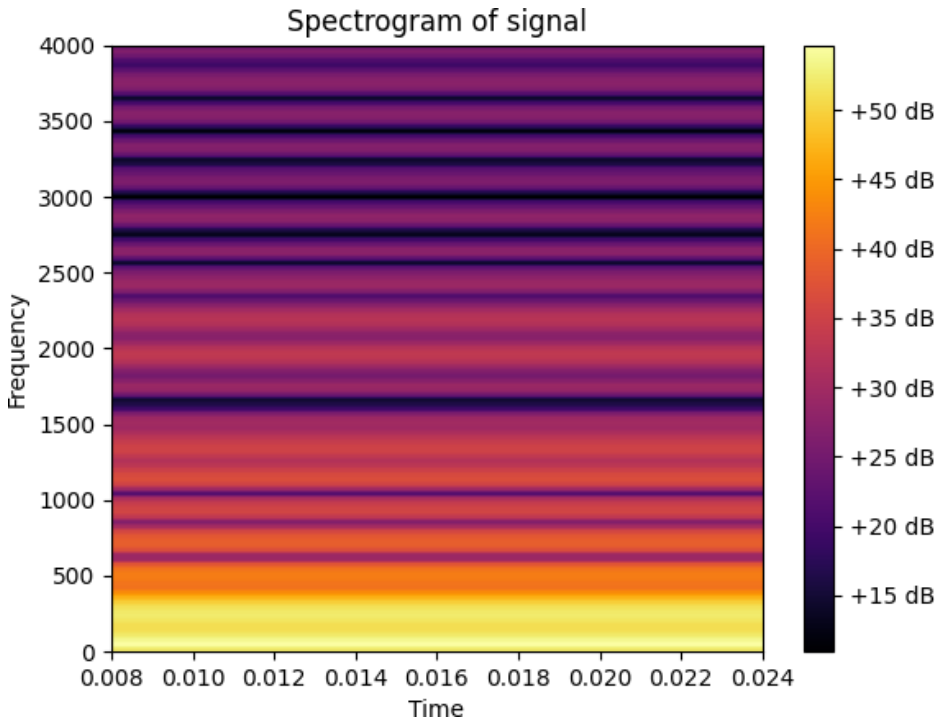
Τα spectrograms είναι η αναπαράσταση του σήματος σε μία εικόνα. Ο άξονας χ δείχνει τον χρόνο ενώ ο Υ την συχνότητα.

Τα spectrograms υπολογίζονται με την συνάρτηση spectrogram του scipy μέσω μετασχηματισμών fourier. Οι εικόνες αυτές είναι τα δεδομένα που θα δώσουμε στο μοντέλο μας έτσι ώστε να πάρουμε την πρόβλεψη για τα ψηφία. Κάθε σήμα(ηχογράφηση) έχει το δικό του spectrogram και συνεπώς επειδή για κάθε ψηφίο υπάρχουν 50 σήματα από διαφορετικούς ομιλητές ο αλγόριθμος ταξινόμησης που θα επιλέξουμε θα συγκλίνει.

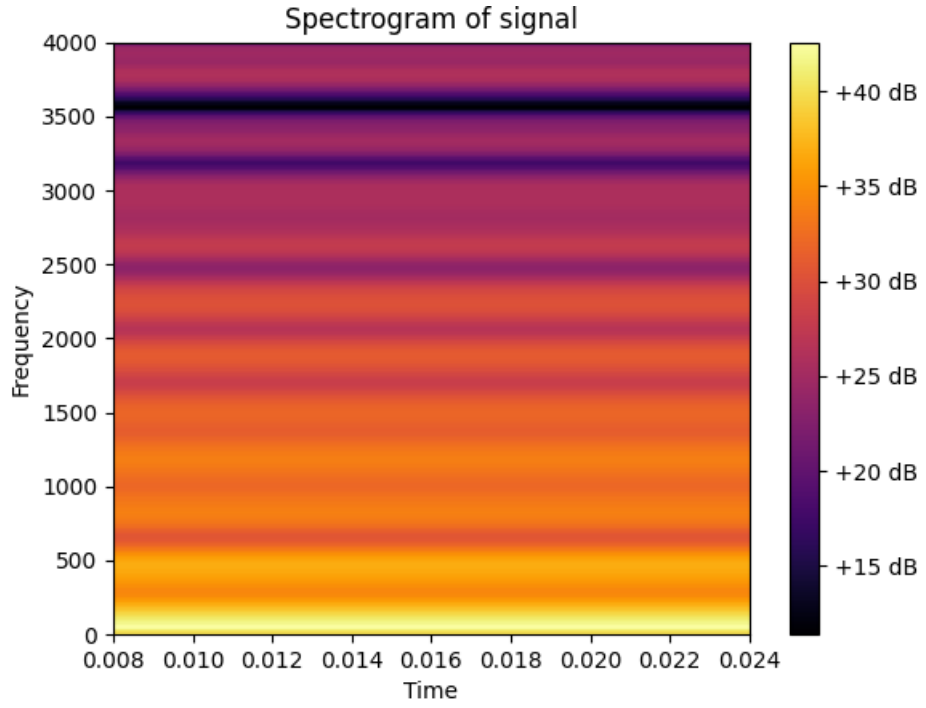
Εμφανίζω για τα 4 δείγματατα τα spectograms τους όπως και παραπάνω.

Τα spectrograms αποθηκεύονται όλα μαζί στη μεταβλητή spec

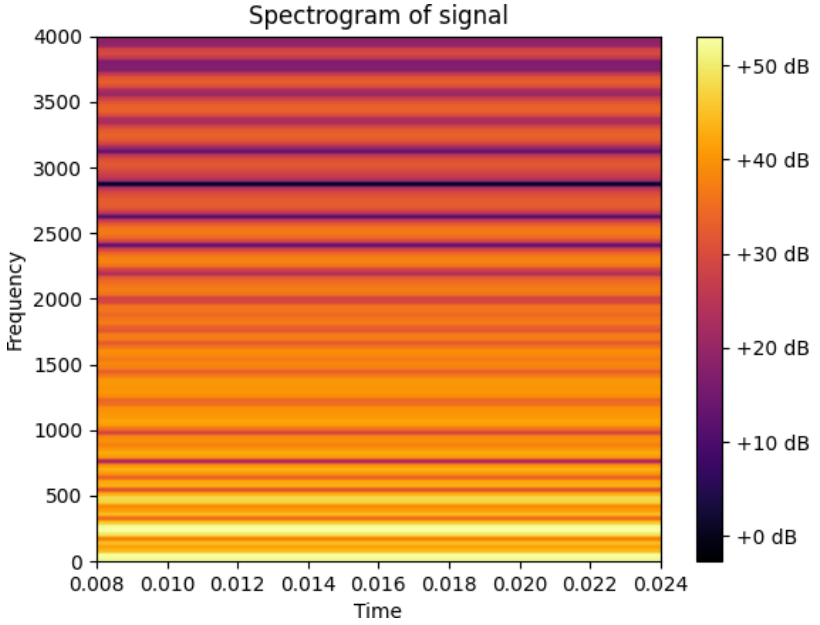
Δείγμα 100



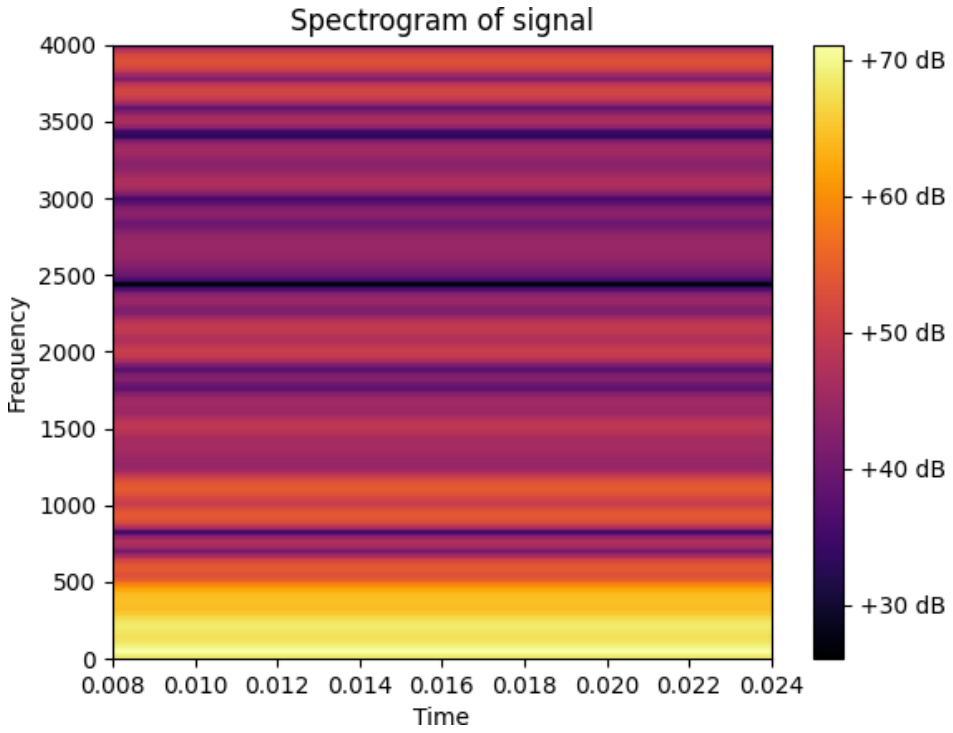
Δείγμα 300



Δείγμα 400



Δείγμα 600



Για όλα τα spectrograms υπολογίζουμε το scaled version τους με τον MinMaxScaler της sklearn. Αυτό γίνεται έτσι ώστε τα δεδομένα να είναι στο ίδιο range για να εφαρμόσουμε το SVM στο επόμενο βήμα.

Πρόβλεψη

Εφόσον έχουμε τα scaled spectrograms τότε χωρίζουμε τα δεδομένα μας σε train και test sets με την συνάρτηση train_test_split της sklearn. Δίνουμε 70% στο train set και 30% στο test set.

Ο ταξινομητής που επιλέξαμε είναι ένας one vs all support vector machines.

Στον πιο απλό τύπο του, το SVM δεν υποστηρίζει ταξινόμηση πολλαπλών κλάσεων. Υποστηρίζει binary ταξινόμηση και διαχωρισμό δεδομένων σε δύο κατηγορίες. Για την ταξινόμηση πολλαπλών κλάσεων, η ίδια αρχή χρησιμοποιείται μετά τη διάσπαση του προβλήματος ταξινόμησης σε πολλαπλά προβλήματα δυαδικής ταξινόμησης.

Το πρόβλημα ταξινόμησης εδώ έχει 10 πιθανές κλάσεις για τα ψηφία από 0 έως 9. Οπότε χρησιμοποιείται η τεχνική one vs all.

Ο ταξινομητής χρησιμοποιεί N*(N-1) / 2 support vectors. Εδώ δηλαδή χρησιμοποιεί 10*9/2 = 45.

Το sklearn παρέχει την κλάση SVM.

Χρησιμοποιούμε το LinearSVC(tol=1e-5). Η συνάρτηση αυτή ορίζει το multi-class classification για το SVM, το tol=1e-5 είναι η παράμετρος που δείχνει το tolerance του αλγορίθμου.

Κάνουμε fit στο X_train και ελέγχουμε για τα δεδομένα του Y_test με την metrics.accuracy_score της sklearn που αθροίζει τα σωστά αποτελέσματα δια το πλήθος όλων των δειγμάτων του Y_test.

Παρακάτω εμφανίζουμε τα αποτελέσματα για τα δείγματα και το γενικό accuracy.

```
Digit on signal 100 is 1 and SVM predicted 1
Digit on signal 300 is 9 and SVM predicted 7
Digit on signal 400 is 4 and SVM predicted 4
Digit on signal 600 is 1 and SVM predicted 1
```

Βλέπουμε πως ο ταξινομητής πετυχαίνει ένα καλό accuracy όπου κατά μέσο όρο βρίσκει περίπου 7/10 ψηφία. Ένα SVM είναι μια απλή αρχιτεκτονική, καλύτερα αποτελέσματα θα μπορούσαν να εξαχθούν είτε από κάποιο νευρωνικό δίκτυο είτε μέσω hidden Markov models όπως στην προσέγγιση που παρέχει το βιβλίο.

```
Total accuracy of SVM on X_test: 0.6832532717619048
```

Resources

Βιβλίο μαθήματος

https://github.com/Jakobovski/free-spoken-digit-dataset

https://scipy.org

https://scikit-learn.org/stable/

https://numpy.org

https://matplotlib.org