Πανεπιστήμιο Πειραιά

Τμήμα Πληροφορικής

Εργασία στο μάθημα **Επεξεργασία Σημάτων Φωνής και Ήχου**

Αριστοτέλης Ματακιάς Α.Μ. : Π19100

Ιούνιος 2023

Περιεχόμενα

1 Ορισμός του Προβλήματος

Ζητούμενα

Καλείστε να υλοποιήσετε ένα ASR σύστημα, που δέχεται είσοδο μία ηχογράφηση κάθε φορά, η οποία συνιστά πρόταση αποτελούμενη από 5-10 ψηφία της Αγγλικής γλώσσας που έχουν ειπωθεί με αρκούντως μεγάλα διαστήματα παύσης.

Το σύστημα προχωρά στην κατάτμηση της πρότασης σε λέξεις χρησιμοποιώντας υποχρεωτικά έναν ταξινομητή background vs foreground της επιλογής σας. Από τις λέξεις που προκύπτουν, υπολογίστε τη θεμελιώδη συχνότητα του ομιλητή.

Στη συνέχεια, το σύστημα αναγνωρίζει κάθε λέξη χρησιμοποιώντας ως φασματική αναπαράσταση μόνο το απλό φασματογράφημα (επιλέξτε φασματική περιοχή αν θέλετε). Αν χρειαστείτε δεδομένα εκπαίδευσης, χρησιμοποιήστε μόνο δημόσια σύνολο(α) δεδομένων από το Internet. Δεν χρειάζεται να τα συμπεριλάβετε στα παραδοτέα αλλά μόνο να περιγράψετε πώς ελήφθησαν.

2 Αλγοριθμική Περιγραφή

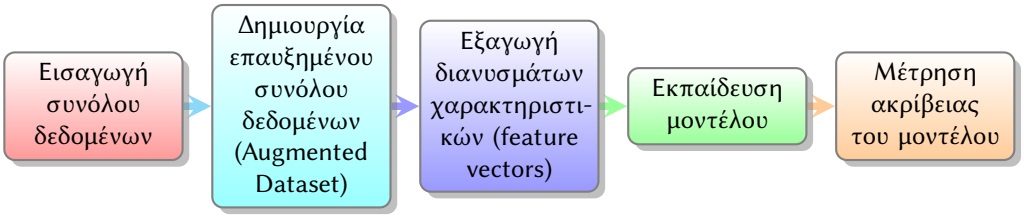
2.1 Εισαγωγή

Παρακάτω δίνεται μία σχηματική αναπαράσταση των αλγοριθμικών διαδικασιών του υλοποιημένου ASR συστήματος με ένα διάγραμμα ροής.

Σχήμα 1: Σχηματικό διάγραμμα του ASR συστήματος

Όπως υποδεικνύει το διάγραμμα το ASR σύστημα δέχεται σαν είσοδο ένα σήμα φωνής και στη συνέχεια εφαρμόζεται μία σειρά ενεργειών με σκοπό να αναγνωριστούν οι λέξεις που υπάρχουν στο σήμα αυτό. Η διαδικασία αυτή γίνεται σειριακά και έχει αναλυθεί στα πιο στοιχειώδη βήματά της ώστε να γίνει πιο εύκολα κατανοητή. Κάθε πλαίσιο του διαγράμματος αποτελεί μία αλγοριθμική διαδικασία, η οποία θα αναλυθεί στις επόμενες υποενότητες.

Ένα επιπλέον ζήτημα που δεν περιλαμβάνεται στο Σχήμα 1 είναι το πώς εκπαιδεύεται το μοντέλο που αναφέρεται στο τελευταίο μπλοκ. Το μοντέλο αυτό είναι ένας Ταξινομητής και εκπαιδεύεται μέσω ενός μεγάλου συνόλου δεδομένων από το δημόσιο audioMNIST dataset, το οποίο είναι διαθέσιμο και σε επίσημο Github repository (https://github.com/soerenab/AudioMNIST) [1].

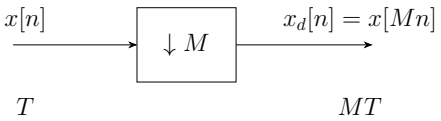


Σχήμα 2: Σχηματικό διάγραμμα της διαδικασίας εκπαίδευσης του Μοντέλου αναγνώρισης λέξεων

Στο σχήμα 2 παρουσιάζονται επιγραμματικά τα βήματα της διαδικασίας εκπαίδευσης του μοντέλου αναγνώρισης λέξεων. Σε επόμενη υποενότητα θα δοθεί μία αναλυτική περιγραφή.

2.2 Αλλαγή ρυθμού δειγματοληψίας

Μία βασική διαδικασία που πρέπει να πραγματοποιηθεί πριν γίνει περαιτέρω επεξεργασία πάνω στο σήμα είναι η αλλαγή του ρυθμού δειγματοληψίας. Πρόκειται για μία κλασσική πράξη που γίνεται στην Επεξεργασία Ήχου και την Επεξεργασία Σήματος γενικότερα. Ο λόγος για τον οποίο εφαρμόζεται είναι να μειωθεί το μήκος του αρχικού σήματος, διευκολύνοντας έτσι στην χρονική πολυπλοκότητα των υπολοίπων διαδικασιών (και κυρίως του ταξινομητή background vs foreground). Αυτό θα μπορούσε να επιτευχθεί με μία απλή πράξη υποδειγματοληψίας. Πράγματι, η υποδειγματοληψία με παράγοντα M στο αρχικό σήμα έχει ως αποτέλεσμα τη δημιουργία ενός νέου σήματος του οποίου η περίοδος δειγματοληψίας είναι $T' = MT$ (όπου T η περίοδος δειγματοληψίας του αρχικού σήματος). Αυτό συνοψίζεται στο σχήμα 3.



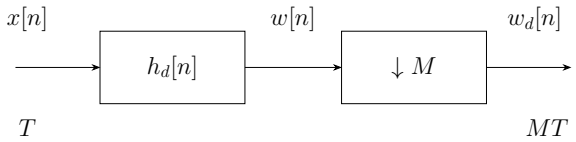
Σχήμα 3: Μπλοκ διάγραμμα αναπαράστασης υποδειγματοληψίας

Ωστόσο, η υποδειγματοληψία προκαλεί παραμόρφωση του αρχικού σήματος, εκτός αν εφαρμοστεί ένα κατάλληλο κατωπερατό φίλτρο πριν την υποδειγματοληψία, ή εκτός αν ο αρχικός ρυθμός δειγματοληψίας είναι διπλάσιος του ρυθμού Nyquist.

Προκειμένου να μειώσουμε τον ρυθμό δειγματοληψίας ενός σήματος διακριτού χρόνου κατά παράγοντα M χωρίς παραμόρφωση πρέπει να εξασφαλιστεί ότι η υψηλότερη συχνότητα που εμφανίζεται στο σήμα διακριτού χρόνου δεν είναι μεγαλύτερη από $\frac{Fs}{2M}$. Αυτό επιτυγχάνεται αν το αρχικό σήμα φιλτραριστεί αρχικά από ένα κατωπερατό φίλτρο, όπως αναφέραμε παραπάνω, του οποίου η κρουστική απόκριση είναι:

$$H(e^{j\omega}) = \begin{cases} 1, & \text{αν } |\omega| < \frac{\pi}{M}. \\ 0, & \text{αν } \frac{\pi}{M} < |\omega| \leq \pi. \end{cases}$$

Η συχνότητα αποκοπής του φίλτρου αυτού είναι $\omega_c = \frac{\pi}{M}$, η οποία ουσιαστικά αντιστοιχεί στη συχνότητα $\frac{F_s}{2M}$, επειδή ακριβώς $\frac{2\pi T F_s}{2M} = \frac{\pi}{M}$. Αρκεί λοιπόν το φιλτραρισμένο σήμα να υποδειγματοληπτηθεί με ρυθμό M. Η διαδικασία συνοψίζεται στο σχήμα 4.



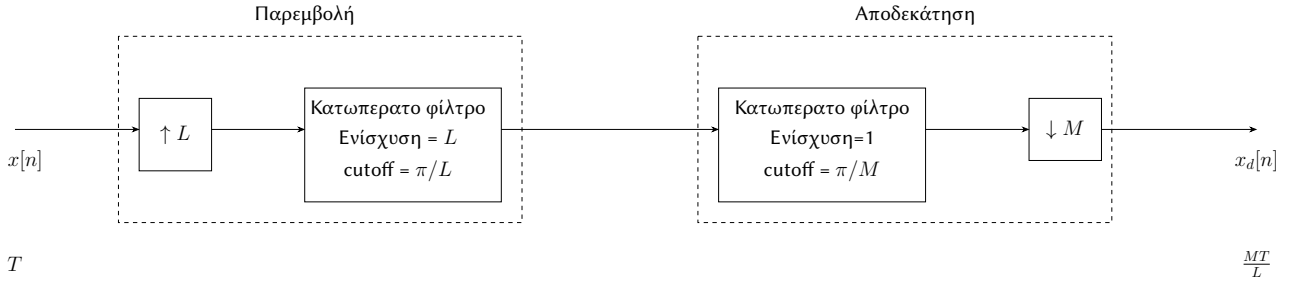
Σχήμα 4: Μπλοκ διάγραμμα αναπαράστασης της αποδεκάτισης με κατωπερατό φίλτρο

Η διαδικασία αυτή ωστόσο αφορά την μείωση ρυθμού δειγματοληψίας κατά έναν ακέραιο παράγοντα. Στην περίπτωση του δικού μας συστήματος, έχουμε σαν αρχική συχνότητα τιμές 44100Hz και 20000Hz και θέλουμε να τις μειώσουμε στα 8000Hz. Ο ρυθμός δειγματοληψίας θα πρέπει να μειωθεί κατά $8000/44100 = 80/441 = 0.1814$ και $8000/20000 = 4/10 = 0.4$. Στις περιπτώσεις αυτές δεν μπορούμε να βασιστούμε στη παραπάνω μέθοδο καθώς σε αυτή τη περίπτωση θέλουμε η αναλογία της αρχικής και της τελικής συχνότητας δειγματοληψίας να εκφράζεται από έναν ρητό αριθμό. Αυτό το πρόβλημα εμφανίζεται αρκετές φορές στην επεξεργασία ήχου, καθώς οι συχνότητες που χρησιμοποιούνται αρκετά είναι οι 44.1kHz και 48kHz οι οποίες διαφέρουν κατά παράγοντα 1.088. Υπάρχει λοιπόν η ανάγκη μη ακέραιας μετατροπής ρυθμού δειγματοληψίας (Non-Integer Sample Rate Conversion) και θα παρουσιάσουμε παρακάτω δύο γνωστές μεθόδους που υπάρχουν.

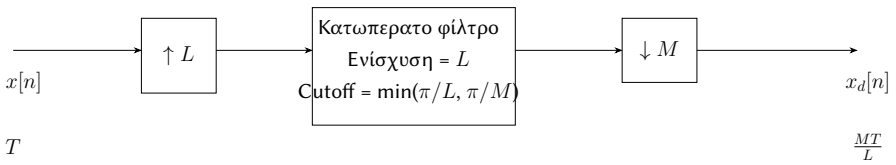
2.2.1 Μέθοδοι μη ακέραιας μετατροπής ρυθμού δειγματοληψίας

Σύμφωνα με τα βιβλία των Oppenheim [2] και Lyons [3] υπάρχουν δύο προσεγγίσεις στο πρόβλημα αυτό:

- Έστω ότι η αναλογία του αρχικού και του τελικού ρυθμού δειγματοληψίας είναι L/M (ή M/L αν μιλάμε για περίοδο δειγματοληψίας). Πραγματοποιούμε παρεμβολή κατά παράγοντα L και στη συνέχεια αποδεκάτιση κατά παράγοντα M και το αποτέλεσμα θα είναι το τελικό σήμα να έχει τον επιθυμητό ρυθμού δειγματοληψίας. Η λύση αυτή υπάρχει και στο βιβλίο του Rabiner [4].



Σχήμα 5: Μπλοκ διάγραμμα συστήματος μετατροπής ρυθμού δειγματοληψίας κατά ρητό παράγοντα με δύο φίλτρα



Σχήμα 6: Μπλοκ διάγραμμα συστήματος μετατροπής ρυθμού δειγματοληψίας κατά ρητό παράγοντα με ένα φίλτρο

Όπως φαίνεται στο σχήμα 5 το σύστημα για την αλλαγή της περιόδου δειγματοληψίας κατά παράγοντα M/L θα αποτελείται από μία σύνδεση σε σειρά ενός συστήματος παρεμβολής και ενός συστήματος αποδεκάτισης. Αυτό θα έχει ως αποτέλεσμα τη διαδοχή του φίλτρου παρεμβολής από το φίλτρο αποδεκάτισης. Τα δύο διαδοχικά φίλτρα μπορούν να αντικατασταθούν από ένα όπως φαίνεται στο σχήμα 6. Το φίλτρο αυτό θα έχει συχνότητα αποκοπής το ελάχιστο των π/L και π/M και ενίσχυση L.

Η μέθοδος αυτή όμως έχει μεγάλη υπολογιστική πολυπλκότητα και θα είναι αρκετά αργή (όπως θα φανεί και στην υλοποίηση) για σήματα που δεν έχουν πολύ μικρό μήκος. Αυτό είναι αναμενόμενο καθώς γίνεται παρεμβολή μηδενικών, αυξάνοντας αρκετά το μήκος του σήματος, και στη συνέχεια αποδεκάτιση. Θα μπορούσαμε να πούμε πως αυτή η μέθοδος δεν είναι πολύ αποδωτική γιατί μηδενικά που αρχικά προστίθενται στην συνέχεια αφαιρούνται.

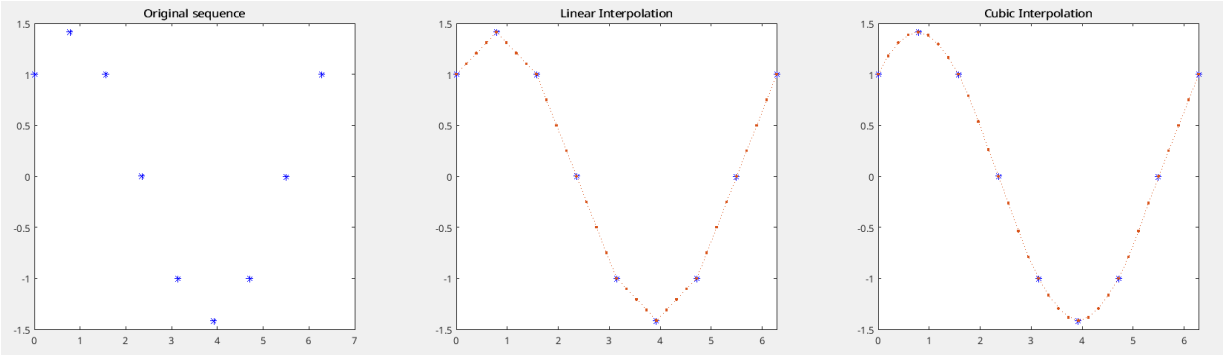
- Ένας άλλος τρόπος που μπορεί να προσεγγιστεί το πρόβλημα της αλλαγής ρυθμού δειγματοληψίας είναι να αντιμετωπιστούν τα δείγματα του σήματος σαν γεωμετρικά σημεία και να δημιουργηθεί το νέο σήμα με μαθηματική παρεμβολή (interpolation). Μπορούμε να επιλέξουμε μέθοδο παρεμβολής ανάλογα με το πόσο γρήγορη ή πόσο ακριβές θέλουμε να είναι το αποτέλεσμα. Στο σχήμα 7 φαίνεται ένα παράδειγμα γραμμικής και κυβικής παρεμβολής για μία ακολουθία σημείων.

2.3 Υψιπερατό φίλτρο FIR

Μετά την αλλαγή ρυθμού δειγματοληψίας το σήμα θα περάσει μέσα από ένα FIR φίλτρο. Το φίλτρο αυτό χρησιμοποιείται για να εξαλείψει την DC συνιστώσα στο αρχικό σήμα. Λέμε ότι ένα σήμα έχει DC συνιστώσα όταν το μέσο πλάτος της καμπύλης του δεν είναι μηδέν. Μια κανονική φωνή είναι ένα συμμετρικό ημιτονικό σήμα, όπου η υψηλή κορυφή ισούται με τη χαμηλή κορυφή. Σε περίπτωση που υπάρχει μετατόπιση DC, το ημιτονικό σήμα δεν είναι συμμετρικό και ο μέσος όρος στο χρόνο δεν είναι μηδέν. Επιπλέον, το σήμα δεν είναι μηδενικό ούτε κατά τις περιόδους σιωπής.

Το φίλτρο έχει τις εξής προδιαγραφές (δεδομένου ότι το σήμα θα έχει πλέον συχνότητα δειγματοληψίας 8.000 Hz):

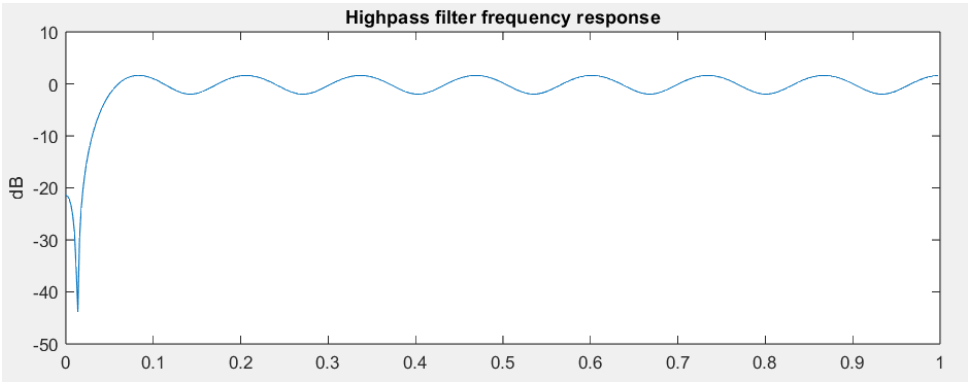
- Ζώνη αποκοπής στο διάστημα $0 \leq |F| \leq 100$ Hz



Σχήμα 7: Παράδειγμα παρεμβολής στο Matlab

- Ζώνη μετάβασης στο διάστημα $100 \leq |F| \leq 200$ Hz
- Ζώνη διέλευσης στο διάστημα $200 \leq |F| \leq 4000$ Hz

Στο σχήμα 8 δίνεται η απόκριση συχνότητας του φίλτρου.



Σχήμα 8: Απόκριση συχνότητας του υψιλοπερατού φίλτρου

2.4 Ταξινομητής background vs foreground

Το αρχικό σήμα ήχου έχει υποστεί αλλαγή ρυθμού δειγματοληψίας και έχει φιλτραριστεί από ένα FIR φίλτρο. Σειρά έχει η διαδικασία εξαγωγής των τμημάτων του σήματος φωνής που έχουν τις λέξεις. Αυτό επιτυγχάνεται με επεξεργασία βραχέως χρόνου, όπου το σήμα χωρίζεται σε επικαλυπτόμενα παράθυρα (frames) και ένας ταξινομητής αποφασίζει για κάθε παράθυρο αν είναι ήχος υποβάθρου (background) ή φωνής (foreground), γι' αυτό και ονομάζεται ταξινομητής background vs foreground.

Πιο αναλυτικά, ο background vs foreground ταξινομητής βασίστηκε στην άσκηση 10.4 του βιβλίο του Rabiner [4]. Υπολογίζεται το zerocrossing rate και η ενέργεια και μετά γίνεται μία διαδικασία κατωφλίωσης για κάθε ένα πλαίσιο. Το πλεονέκτημα αυτού του τρόπου σχεδίασης είναι ότι ο ταξινομητής είναι απλός σε σχεδίαση και δεν απαιτεί ουσιαστικά κάποια εκπαίδευση (άρα δεν έχει ανάγκη και από κάποιο σύνολο δεδομένων).

Πιο αναλυτικά ο ταξινομητής πραγματοποιεί τα εξής βήματα:

1. Επεξεργασία βραχέως χρόνου του σήματος με παράθυρο μήκους 30msec και hop size 10msec. Στο τέλος γίνεται zeropadding εφόσον αυτό είναι απαραίτητο.

2. Υπολογισμός λογαριθμικής ενέργειας και Zerocrossing rate για κάθε παράθυρο με τους παρακάτω τύπους.

$$E_n = 10 \log_{10} \left(\sum_{m=n-L+1}^{n} (x(n)w(n-m))^2 \right)$$

$$Z_n = \frac{R}{2L} \sum_{m=n-L+1}^{n} |sgn(m) - sgn(m-1)| w(n-m)$$

Όπου L, R το μήκος του παραθύρου και η μεταβολή του παραθύρου αντίστοιχα και w παράθυρο hamming.

3. Υπολογισμός της μέσης τιμής και της τυπικής απόκλισης της λογαριθμικής ενέργειας και Zerocrossing rate για τα πρώτα 10 πλαίσια, υποθέτοντας ότι αντιστοιχούν σε υπόβαθρο. Οι τιμές αυτές θα ονομάζονται eavr, esig, zcavg και zcsig.

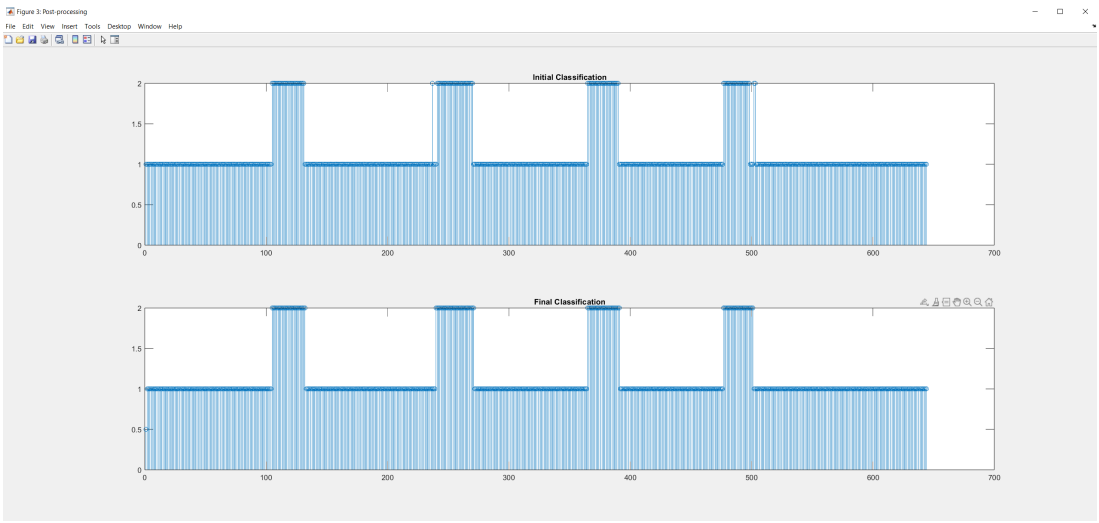
4. Υπολογίζουμε ορισμένες παραμέτρους που έχει ορίσει ο Rabiner:

   - $IZCT = max(IF, zcavg + 3 \times zcsig)$ - Μεταβλητό κατώφλι για το Zerocrossing rate με βάση τη στατιστική ανάλυση του σήματος.
   - $IMX = max(Energy)$ - Απόλυτο πλάτος της μέγιστης κορυφής της παραμέτρου της λογαριθμικής ενέργειας
   - $ITU = IMX - 20$ - Υψηλό κατώφλι για την παράμετρο της λογαριθμικής ενέργειας.

5. Η ταξινόμηση για κάθε παράθυρο i γίνεται ως εξής:

   Αν
   $$(Energy(i) \geq ITU) \&\& (ZeroCrossRate(i) \leq IZCT)$$
   τότε το πλαίσιο i είναι πλαίσιο foreground (class 2), αλλιώς είναι πλαίσιο background (class 1).

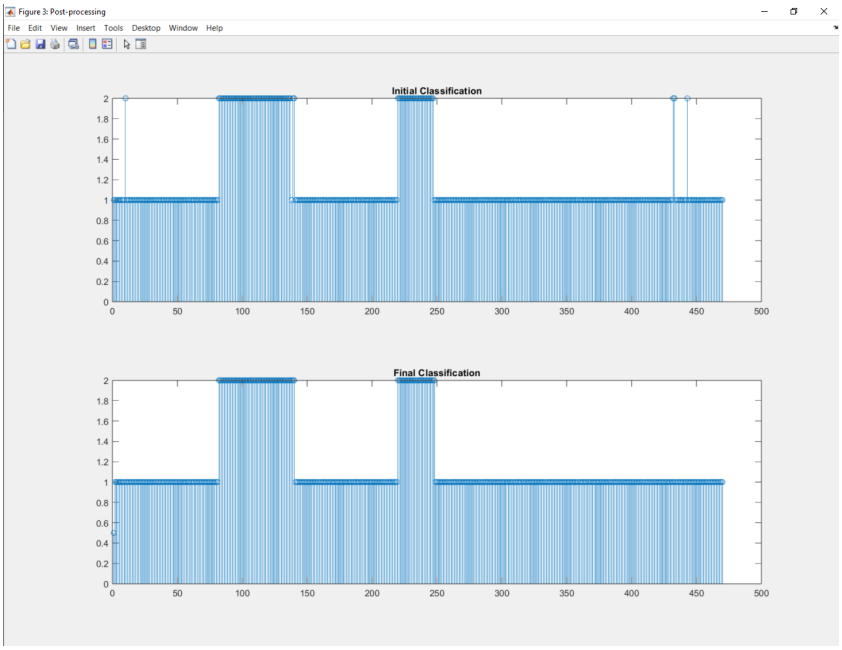
6. Η ταξινόμηση που προκύπτει ενδεχομένως να έχει κάποια σφάλματα. Ένα ενδεχόμενο είναι ένα τμήμα που αντιστοιχεί σε κομμάτι ομιλίας να ανιχνευτεί σαν ξεχωριστά μέρη, όπως φαίνεται στο σχήμα 9. Ένα άλλο ενδεχόμενο είναι μικροί κρότοι που υπάρχουν στο σήμα λόγω θορύβου να ανιχνευτούν από τον ταξινομητή ενώ δεν πρέπει (σχήμα 10).



Σχήμα 9

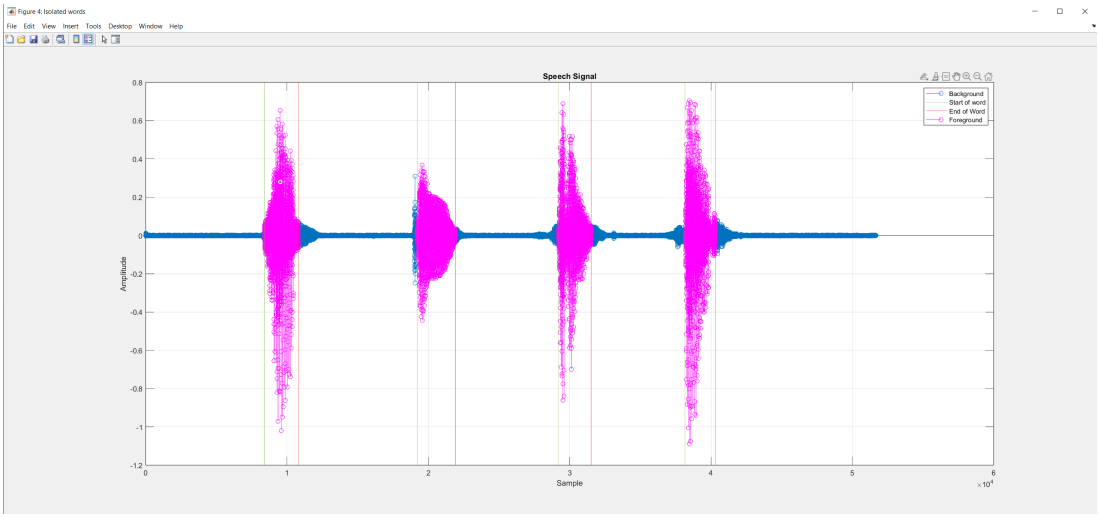
Το πρώτο πρόβλημα μπορεί να λυθεί αν απλά περάσουμε την ακολουθία ταξινόμησης από ένα φίλτρο μεσσαίας τιμης (media filter) και έτσι θα βοηθήσει να εξαφανιστούν μικρά κενά μέσα σε εμφωνα τμήματα (σχήμα 9 κάτω μέρος).

Το δεύτερο πρόβλημα μπορεί να λυθεί αν απλά δώσουμε στον ταξινομητή ένα κατώφλι και του πούμε να αγνοήσει τα τμήματα που είναι πολύ μικρά σε διάρκεια πλαισίων (σχήμα 10 κάτω μέρος). Η σύμβαση αυτή είναι καλή, καθώς σκοπός του ταξινομητή είναι να εντοπίσει λέξεις, και δεν πρόκυται να έχουν πολύ μικρή διάρκεια.



Σχήμα 10

7. Τέλος ο ταξινομητής θα υπολογίσει ποια δείγματα του αρχικού σήματος αντιστοιχούν σε τμήματα φωνής. Αυτό γίνεται με μία απλή σάρωση και θεωρείται ότι μία λέξη θα αποτελείται από συνεχή πλαίσια.



Σχήμα 11

Στο σχήμα 11 φαίνεται η έξοδος του ταξινομητή ουσιαστικά. Με μπλε χρωματίζονται δείγματα υποβάθρου και με μοβ δείγματα φωνής. Η πράσινη κάθετη γραμμή δηλώνει αρχή μίας λέξης και η κόκκινη κάθετη φωνή δηλώνει το τέλος της λέξης.

Ο ταξινομητής σε γενικές γραμμές δουλεύει ικανοποιητικά. Το στοιχείο που πρέπει να εστιάσουμε είναι ότι ο ταξινομητής ”κόβει” ουσιαστικά κάθε λέξη δεξιά και αριστερά. Αυτό σημαίνει ότι οι λέξεις που ανιχνεύονται θα έχουν πάντα διαφορετικό μήκος. Αυτό θα το λύσουμε βάζοντας στο τέλος μηδενικά (zero padding) ώστε όλες οι λέξεις να αποκτούν ίδιο μήκος.

2.5 Υπολογισμός θεμελιώδους συχνότητας

Για τις ανάγκες της εργασίας έχει επιλεχθεί ένας απλός αλγόριθμος υπολογισμού της θεμελιώδους συχνότητας και βασίζεται στην Αυτοσυσχέτιση (autocorrelation). Ο τύπος της συνάρτησης Αυτοσυσχέτισης

για ένα σήμα x μήκους N δειγμάτων είναι ο εξής:

$$R(k) = \sum_{n=k}^{N-1} x(n)x(n-k)$$

Παρακάτω δίνεται ο αμερόληπτος (unbiased) τύπος:

$$R(k) = \frac{1}{N-k} \sum_{n=k}^{N-1} x(n)x(n-k)$$

Η αυτοσυσχέτιση μπορεί να κανονικοποιηθεί με μέγιστη τιμή 1:

$$\gamma(k) = \frac{R(k)}{R(0)}$$

Αυτόν το τύπο θα χρησιμοποιήσουμε για την υπολογισμό της αυτοσυσχέτιση του σήματος. Εδώ πρέπει να επισημάνουμε ότι ο δείκτης k στην αυτοσυσχέτιση συνδέεται με τη συχνότητα μέσω του τύπου:

$$k = \frac{F_s}{F}$$

Επειδή θεωρούμε ότι τα σήματα ήχου θα περιέχουν ανθρώποινη φωνή, μπορούμε σαν σύμβαση να θεωρήσουμε ότι η θεμελιώδη συχνότητα θα είναι στο διάστημα $[F_min, F_max] = [90Hz, 600Hz]$. Τα $90Hz$ αντιστοιχούν σε μία πολύ μπάσα φωνή και τα $600Hz$ σε πολύ ψηλή φωνή.

Δεδομένου αυτού του διαστήματος βρίσκουμε τα αντίστοιχα k:

$$k_1 = \frac{F_s}{600} = \frac{8000}{600} = 13$$

$$k_2 = \frac{F_s}{90} = \frac{8000}{90} = 88$$

Θα υπολογίσουμε λοιπόν τη συνάρτηση αυτοσυσχέτισης για τα k μέσα στο $[13, 88]$ και το k που αντιστοιχεί με τη μεγαλύτερη αυτοσυσχέτιση θα αντιστοιχεί στη θεμελιώδη συχνότητα, και θα βρεθεί με το τύπο $F = Fs/k$.

2.6 Φασματική ανάλυση λέξεων

Μέχρι στιγμής εξετάζαμε το σήμα φωνής στον χρόνο. Προκειμένου όμως να εξάγουμε ένα διάνυσμα χαρακτηριστικών από κάθε λέξη που έχει ανιχνευθεί στο σήμα θα χρησιμοποιήσουμε το φασματογράφημα. Πιο συγκεκριμένα, στο πρόγραμμα χρησιμοποιείται φασματογράφημα παράθυρα Hamming μήκους 100 δειγμάτων, επικάλυψη 80 δειγμάτων και το φάσμα αποτελείται από $\lfloor 100/2 + 1 \rfloor = 51$ συχνότητες. Η συχνότητα δειγματοληψίας θα είναι 8.000 Hz (επειδή μέσω συνάρτησης όλα τα σήματα θα υποστούν αλλαγή ρυθμού δειγματοληψίας), το μήκος του παραθύρου είναι 12.5msec. Θεωρούμε λοιπόν ότι το φασματογράφημα είναι στενής ζώνης (narrowband spectrogram) επειδή το μήκος του παραθύρου είναι μεγαλύτερο του $2T_s = 2/8000 = 0.00025 = 0.25msec$. Στο σχήμα 12 φαίνεται το φασματογράφημα σήματος που αντιστοιχεί στη λέξη 'zero'.

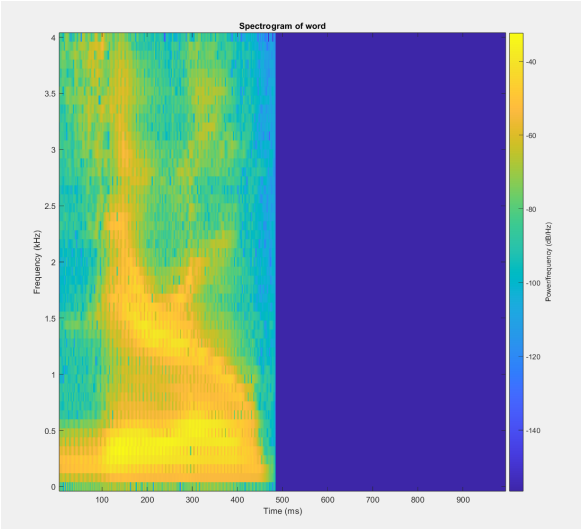
Προκειμένου να εξάγουμε ένα διάνυσμα χαρακτηριστικών απο το φασματογράφημα κάνουμε τα εξής βήματα:

1. Κάνουμε zero padding στο σήμα μέχρι να φτάσει τα 8.000 δείγματα (παρατηρήθηκε στο σύνολο δεδομένων ότι λέξη με τη μεγαλύτερη διάρκεια ήταν περίπου 7.000 δείγματα), δηλαδή 1 δευτερόλεπτο, καθώς $F_s = 8000Hz$

2. Παίρνουμε το φασματογράφημα του σήματος, θα είναι ένας δισδιάστατος πίνακας διαστάσεων 51×396 μιγαδικών αριθμών.

3. Αντικαθιστούμε κάθε μιγαδικό αριθμό με το μέτρο του (απόλυτη τιμή).

4. Μετατρέπουμε τον πίνακα από πίνακα δύο διαστάσεων σε πίνακα μίας διάστασης (flatten). Αυτό είναι το διάνυσμα χαρακτηριστικών που αντιστοιχεί στη λέξη.



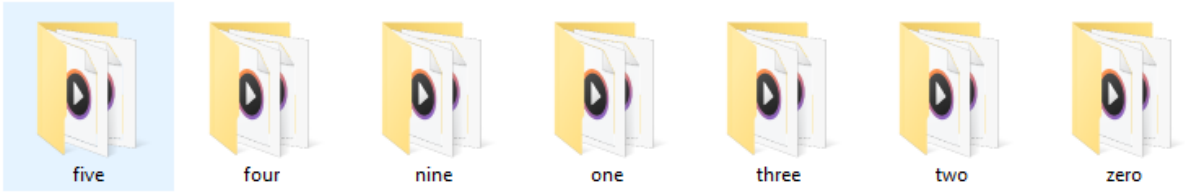
Σχήμα 12: Φασματογράφημα της λέξης zero

2.7 Μοντέλο αναγνώρισης λέξεων

Το μοντέλο που έχει επιλεχθεί για να αναγνωρίζει τις λέξεις στο σήμα φωνής είναι Μηχανή Διανυσματικής Στήριξης (Support Vector Machine). Το μοντέλο δέχεται τα διανύσματα χαρακτηριστικών, τα οποία προκύπτουν από τη φασματική ανάλυση της προηγούμενης ενότητας, καθώς και τις αντίστοιχες ετικέτες και λύνει ένα πρόβλημα ταξινόμησης τάξεων.

Το audioMNIST dataset αποτελείται από 60 φακέλους, όπου κάθε ένας αντιστοιχεί σε έναν άνθρωπο και μέσα στο φάκελο υπάρχουν δείγματα φωνής για όλους τους αριθμούς. Θα αλλάξουμε αυτή την αρχειοθέτηση ώστε να έχουμε φακέλους που περιέχουν δείγματα φωνής ανά αριθμό.

Έχουμε επιλέξει τους αριθμούς **zero**, **one**, **two**, **three**, **four**, **five**, **nine**, οπότε θα δημιουργήσουμε 7 φακέλους και θα βάλουμε μέσα στον καθένα όλα τα αντίστοιχα αρχεία φωνής του audioMNIST dataset (σχήμα 14).



Σχήμα 13

Πριν ξεκινήσουμε τη διαδικασία της εκπαίδευσης πρέπει να πραγματοποιήσουμε ορισμένα βήματα που αναλύονται στις επόμενες υποενότητες.

2.7.1 Δημιουργία επαυξημένου συνόλου δεδομένων (Augmented Dataset)

Το αρχικό σύνολο δεδομένων αποτελείται από .wav αρχεία. Αρχικά κάθε σήμα θα υποστεί αλλαγή ρυθμού δειγματοληψίας στα 8.000 Hz και στη συνέχει θα φιλτραλιστεί με το FIR φίλτροπου έχουμε παρουσιάσει. Στη συνέχεια κάθε σήμα θα περάσει μέσα από τον ταξινομητή foreground vs background. Ιδανικά περιμένουμε πως για κάθε σήμα ο ταξινομητής θα εξάγει μόνο μία λέξη, η οποία είναι και η επιθυμιτή (στη συγκεκριμένη περίπτωση ένας αριθμός). Βέβαια όπως γνωρίζουμε ο ταξινομητής δεν είναι βέλτιστος και ορισμένα σήματα θα έχουν λίγο θόρυβο ή κακή ποιότητα, με αποτέλεσμα σε κάποια σήματα ο ταξινομητής να μην βρίσκει ακριβώς το τμήμα που αντιστοιχεί στη λέξη ή να βρίσκει πάνω από ένα τμήμα φωνής.

Σαν σύμβαση θα θεωρήσουμε ότι αν ο ταξινομητής βρει πάνω από 1 λέξη σε ένα σήμα φωνής τότε θα αγνοήσουμε αυτό το σήμα εξ ολοκλήρου. Αντίθετα αν ο ταξινομητής ανιχνεύσει ακριβώς μία λέξη, τότε θεωρούμε πως αυτή τη λέξη ανιχνεύθηκε σωστά και αποθηκεύουμε το τμήμα που ανιχνεύθηκε.

Μέτα από αυτά τα βήματα θα έχουμε ένα νέο σύνολο δεδομένων που αποκαλείται επαυξημένο (augmented). Σημειώνεται ότι λόγω της απόρριψης που κάναμε όταν ο ταξινομητης βρίσκει πάνω από μία λέξη, το επαυξημένο σύνολο δεδομένων θα έχει χάσει ορισμένα από τα σήματα. Θεωρούμε πως αυτό δεν είναι σημαντική απώλεια καθώς στην υλοποίηση παρατηρήθηκε ότι ο αριθμός απορρίψεων είναι μικρός και το σύνολο δεδομένων εξακολουθεί και είναι επαρκώς μεγάλο.

2.7.2 Εξαγωγή Διανυσμάτων χαρακτηριστικών

Έχοντας δημιουργήσει το επαυξημένο σύνολο δεδομένων, πρέπει να εξάγουμε τα διανύσματα χαρακτηριστικών, γιατί αυτά είναι που θα δωθούν στον ταξινομητή σαν είσοδος. Υπενθυμίζεται ότι το επαυξημένο σύνολο δεδομένων αποτελείται από σήματα φωνής που έχουν συχνότητα δειγματοληψίας 8000 Hz, έχουν φιλτραριστεί και οι λέξεις έχουν ”κοπεί” δεξία και αριστερά, δηλαδή κάθε σήμα αποτελείται μόνο από την λέξη που της αντιστοιχεί χωρίς καμία παύση. Αυτό σημαίνει ότι τα σήματα έχουν όλα διαφορετικό μήκος, ανάλογα με το πόσο χρόνο διαρκεί η κάθε λέξη. Παρατηρήθηκε στην υλοποίηση ότι μεγαλύτερο μήκος στο σύνολο δεδομένων είναι περίπου 7.300 δείγματα και μικρότερο μήκος περίπου 800 δείγματα. Θα κανονικοποιήσουμε τα σήματα κάνοντας zero padding στο τέλος κάθε σήματος μέχρι το μήκος να γίνει 8.000 δείγματα (δηλαδή 1 δευτερόλεπτο επειδή η συχνότητα δειγματοληψίας είναι 8.000 Hz).

Έτσι έχουμε ένα σύνολο σημάτων φωνής με ίδιο μήκος. Μπορούμε λοιπόν να δημιουργήσουμε το φασματογράφημα της κάθε λέξης του συνόλου. Το φασματογράφημα θα έχει ακριβώς τις ίδιες διαστάσεις για όλα τα σήματα. Θα εξάγουμε το πραγματικό μέρος κάθε πεδίου και στη συνέχεια θα μετατρέψουμε το δισδιάστατο διάνυσμα σε μονοδιάστατο και αυτό θα είναι το διάνυσμα χαρακτηριστικών.

2.7.3 Εκπαίδευση μοντέλου

Για να εκπαιδεύσουμε τη Μηχανή Διανυσματικής Στήριξης αρκεί να δώσουμε σαν είσοδο τα διανύσματα χαρακτηριστικών που έχουν υπολογιστεί καθώς και τις ετικέτες που τους αντιστοιχούν. Σημειώνεται ότι δεν θα δώσουμε για εκπαίδευση όλο το σύνολο δεδομένων αλλά ένα μέρος αυτού. Το σύνολο θα χωριστεί σε σύνολο εκπαίδευσης (training set) και σύνολο ελέγχου (testing set) με ποσοστά 80% και 20% αντίστοιχα του αρχικού συνόλου δεδομένων. Το testing set θα χρησιμοποιηθεί για την αξιολόγηση της εκπαίδευσης του μοντέλου, το οποίο θα εξεταστεί στην επόμενη ενότητα.

2.7.4 Αξιολόγηση επίδοσης μοντέλου

Για την αξιολόγηση της επίδοσης του μοντέλου χρησιμοποιούνται οι παρακάτω μετρικές:

- **Ποσοστό σφάλματος (resubstitution error)**

  Το ποσοστό σφάλματος (resubstitution error) είναι ουσιαστικά το συνολικό σφάλμα του ταξινομητή στο training set. Αν το σφάλμα αυτό είναι χαμηλό σημαίνει ότι ο ταξινομήτης είναι καλός αλλά υπάρχει περίπτωση να έχει συμβεί overfitting. Μια καλύτερη ένδειξη θα πάρουμε με το cross-validation error.

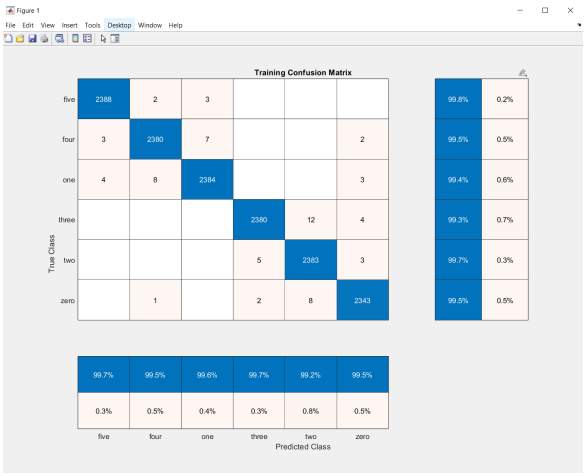
- **10-fold cross-validation** Στο 10-fold cross-validation, το training set χωρίζεται τυχαία σε 10 μέρη και ο ταξινομητής θα εκπαιδευτεί ξεχωριστά για το κάθε ένα και τα σφάλματα θα συνυπολογιστούν. Από το αποτέλεσμα θα καταλάβουμε πόσο καλή ”γενικότητα” έχει πετύχει ο ταξινομητής.

- **Πίνακας σύγχυσης (Confusion matrix)**

  Ο πίνακας σύγχυσης χρησιμοποιείται αρκετές φορές για την αξιολόγηση ταξινομητών. Ουσιαστικά μας δίνει τους ακριβείς αριθμούς προβλέψεων για κάθε κλάση σε ένα σύνολο δεδομένων (Παράδειγμα στο σχήμα 14).

  Επειδή έχουμε διαχωρίσει το αρχικό σύνολο σε training set και testing set θα εμφανίσουμε τους αντίστοιχους πίνακες για το κάθε σύνολο ξεχωριστά.

Σημειώνεται πως θα αναφέρουμε τα αποτελέσματα των μετρικών μας στην ενότητα ’Συμπεράσματα και παρατηρήσεις’.



Σχήμα 14: Παράδειγμα ενός πίνακα σύγχυσης (Confusion matrix)

3 Λεπτομέρειες Υλοποίησης

3.1 Αλλαγή ρυθμού δειγματοληψίας

Στην Αλγοριθμική περιγραφή παρουσιάστηκαν δύο τρόποι για την αλλαγή ρυθμού δειγματοληψίας κατά μη ακέραιο παράγοντα. Στην υλοποίηση υπάρχουν 3 συναρτήσεις:

- Συνάρτηση **resample**: παρέχεται έτοιμη από το Matlab και σύμφωνα με τις οδηγίες της, χρησιμοποιεί τη μέθοδο παρεμβολής και αποδεκάτισης με ένα FIR φίλτρο.
- Συνάρτηση **srcLowPassFilter**: υλοποιήθηκε και περιλαμβάνεται στον κώδικα της εργασίας από την ομάδα. Χρησιμοποιεί την μέθοδο παρεμβολής και αποδεκάτισης με FIR φίλτρο, το οποίο έχει σχεδιαστεί με τη συνάρτηση **fir1** [1].
- Συνάρτηση **srcInterpolation**:υλοποιήθηκε και περιλαμβάνεται στον κώδικα της εργασίας από την ομάδα. Χρησιμοποιεί μαθηματική παρεμβολή, και συγκεκριμένα κυβική, με την συνάρτηση **interp1** του Matlab.

Όλες οι συναρτήσεις δέχονται σαν είσοδο το σήμα, την αρχική συνάρτηση δειγματοληψίας και την επιθυμητή συνάρτηση δειγματοληψίας (σε Hz) και επιστρέφουν το νέο σήμα.

Η συνάρτηση **resample** του Matlab είναι αρκετά γρήγορη, ειδικά σε σχέση με την **srcLowPassFilter** που υλοποιήθηκε. Ενδεχομένως το Matlab να έχει ορισμένες βελτιστοποιήσεις για την **resample**, καθώς η **srcLowPassFilter** χρησιμοποιεί την ίδια μέθοδο, αλλά είναι αρκετά πιο αργή.

Αντίθετά, η **srcInterpolation** είναι αρκετά γρήγορη, δεδομένου ότι βασίζεται στην συνάρτηση **interp1** του Matlab, η οποία πραγματοποιεί γρήγορα τη μαθηματική παρεμβολή.

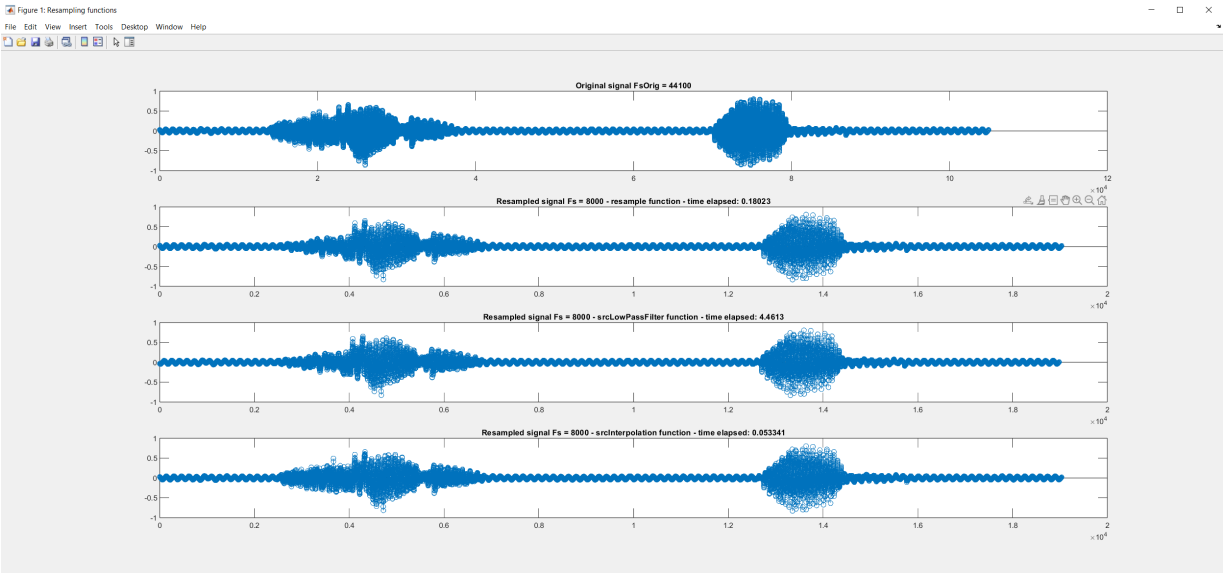
Στο κώδικα περιέχεται το **src_demo_script.m** το οποίο πραγματοποιεί αλλαγή ρυθμού δειγματοληψίας σε ένα σήμα ήχου διάρκειας περίπου δύο δευτερολέπτων. Η αρχικη΄ή συχνότητα δειγματοληψίας είναι 44.1kHz και η επιθυμητή 8kHz.

Το script πραγματοποιεί την αλλαγή χρησιμοποιώντας και τις τρεις συναρτήσεις που παρουσιάστηκαν και υπολογίζει τον χρόνο εκτέλεσης της κάθε μίας. Εμφανίζεται το figure (παράθυρο) που φαίνεται στο σχήμα 15 στο οποίο υπάρχουν: το αρχικό σήμα, το σήμα που προέκυψε από τη **resample**, τη **srcLowPassFilter** και τη **srcInterpolation** και στους τίτλους των υπογραφημάτων εμφανίζεται και ο χρόνος εκτέλεσης της αντίστοιχης συνάρτησης.

Όπως βλέπουμε οι **resample** και η **srcInterpolation** διαρκούν λιγότερο από 1 δευτερόλεπτο, ενώ η **srcLowPassFilter** διαρκεί 4.4 δευτερόλεπτα. Για αυτό το λόγο αποφασίστηκε στο πρόγραμμα να χρησιμοποιηθεί η **srcInterpolation**.

Τέλος το script θα παίξει το αρχικό σήμα και τα σήματα που προέκυψαν, το ένα μετά το άλλο. Επειδή καλύται η συνάρτηση **pause** του Matlab, πρέπει ο χρήστης να πατήσει ένα πλήκτρο για να παίξει επόμενο σήμα. Αυτό έγινε για να μπορεί ο χρήστης να ακούσει εύκολα το κάθε σήμα, χωρίς επικαλύψεις.

[1] Η υλοποίηση βασίστηκε σε ένα άρθρο του Matlab (https://www.mathworks.com/help/signal/ref/resample.html)



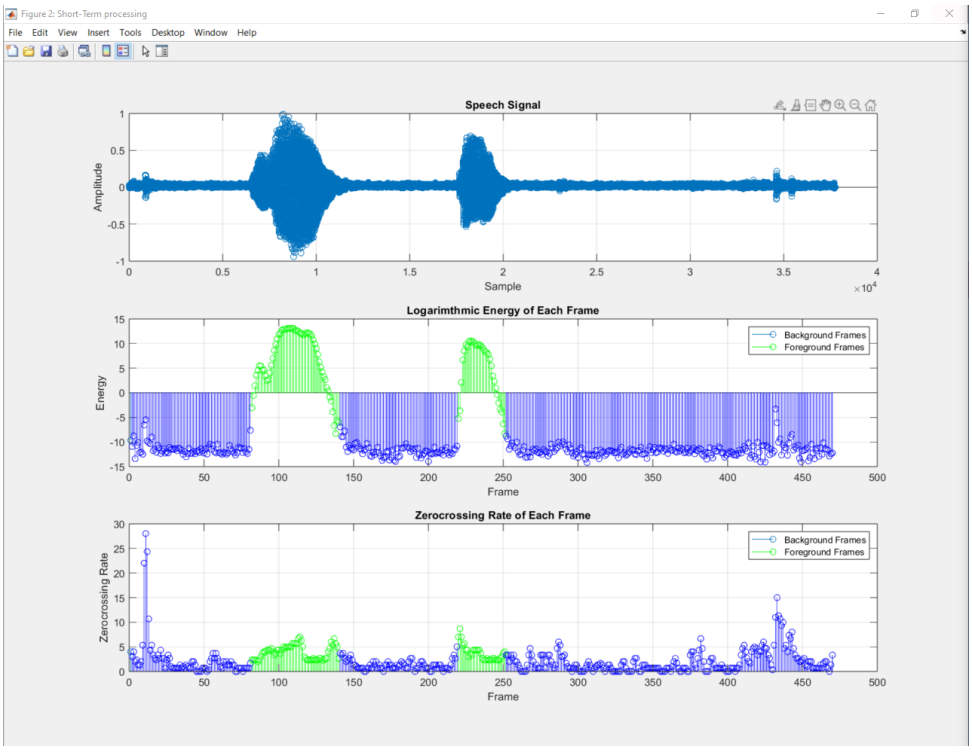
Σχήμα 15: Σύγκριση των συναρτήσεων αλλαγής ρυθμού δειγματοληψίας

3.2 Υψιπερατό φίλτρο FIR

Το υψιλοπερατό φίλτρο FIR έχει υλοποιηθεί με τη συνάρτηση **firpm** του Matlab, η οποία δέχεται τις προδιαγραφές των άκρων ζώνης ιδανικής απόκρισης συχνότητας ανα ζεύγη και κανονικοποιημένες με το 1 να αντιστοιχεί στη μισή συχνότητα δειγματοληψίας (ω/π αντί για $\omega/2\pi$). Συγκεκριμένα οι προδιαγραφές αυτές δίνονται ως το παρακάτω διάνυσμα (όπου $F_s = 8000Hz$):

$$[0, 100, 200, F_s/2]/(F_s/2);$$

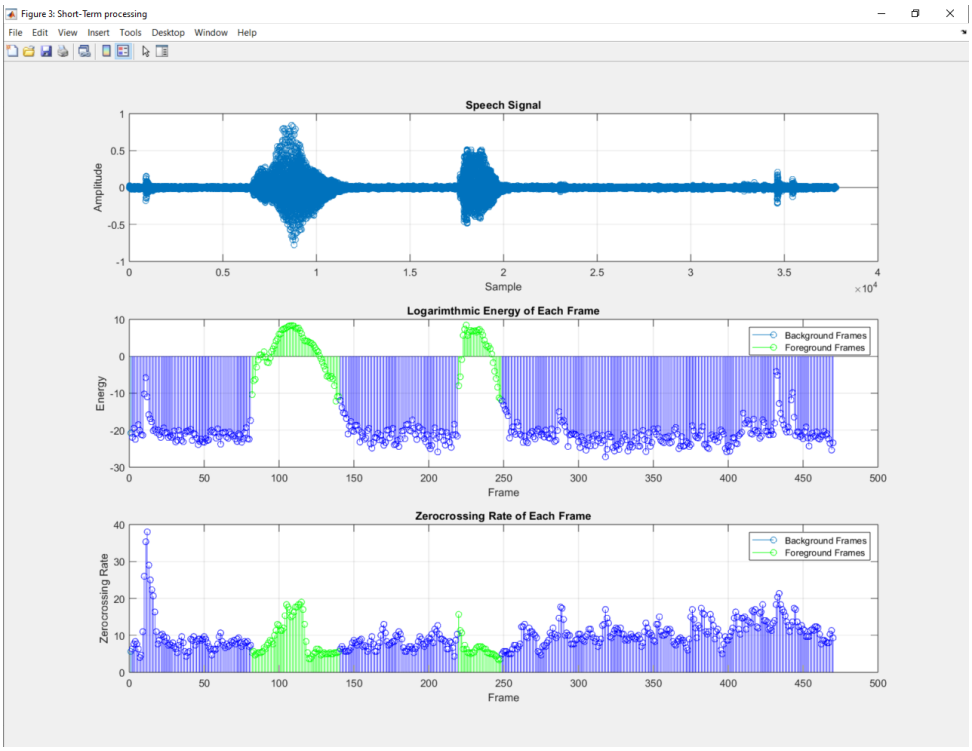
Στα σχήματα 17 και 16 φαίνεται Σήμα φωνής, Ενέργεια βραχέως χρόνου και Zero-crossing rate με το FIR φίλτρο και χωρίς το FIR φίλτρο αντίστοιχα.



Σχήμα 16: Σήμα φωνής, Ενέργεια βραχέως χρόνου και Zero-crossing rate χωρίς το FIR φίλτρο

Γενικά φαίνεται ότι αυξάνεται το πλάτος στην Ενέργεια και στο Zero-crossing rate όταν χρησιμοποιείται

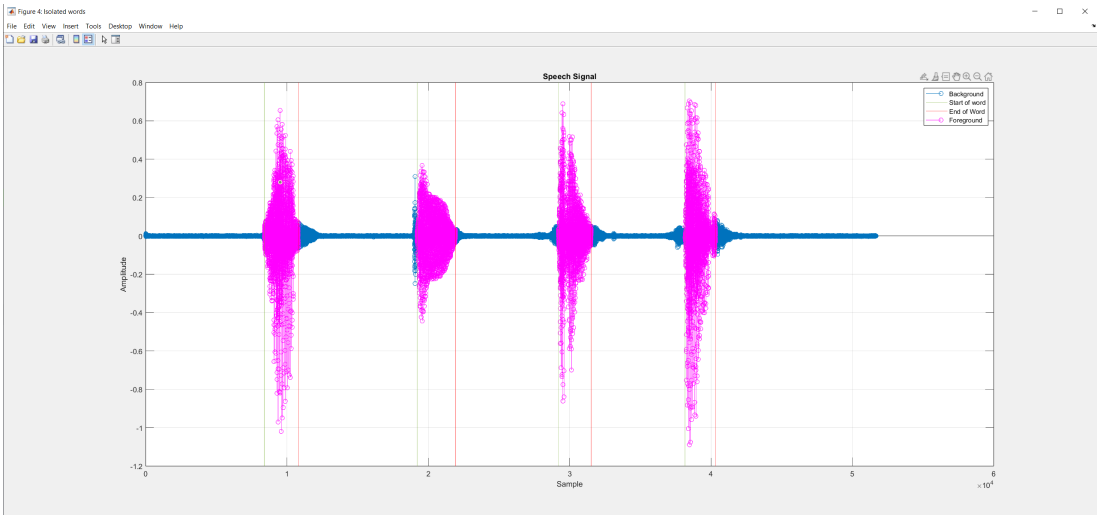
το φίλτρο. Γενικά παρατηρήθηκε ότι ο ταξινομητής background vs foreground πηγαίνει λίγο καλύτερα όταν χρησιμοποιείται το φίλτρο. Επίσης επειδή το φίλτρο κόβει τις χαμηλές συχνότητες, το σήμα ακούγεται πιο ραδιοφωνικό.



Σχήμα 17: Σήμα φωνής, Ενέργεια βραχέως χρόνου και Zero-crossing rate με το FIR φίλτρο

3.3 Ταξινομητής background vs foreground

Ο ταξινομητής που παρουσιάστηκε στην Αλγοριθμική περιγραφή έχει υλοποιηθεί στη συνάρτηση **isolateWords**. Η συνάρτηση αυτή δέχεται σαν είσοδο το σήμα φωνής και επιστρέφει ένα cell array το οποίο περιέχει τα ξεχωριστά σήματα που αντιστοιχούν στις λέξεις. Τα σήματα αυτά ουσιαστικά αντιστοιχούν στα μοβ δείγματα που φαίνονται στην εικόνα 18.



Σχήμα 18

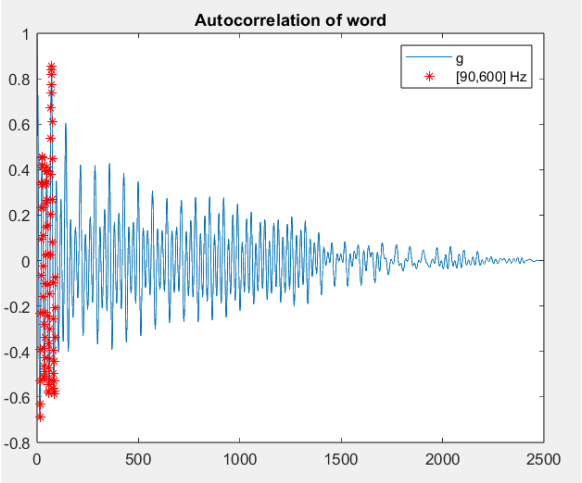
Η συνάρτηση **isolateWords** δεν πραγματοποιεί κάποια πράξη επεξεργασίας σήματος (υποδειγματοληψια, φιλτράρισμα ή άλλα) και απλά δουλεύει στο σήμα που δίνεται. Θα πρέπει πριν δοθεί σαν είσοδος να γίνουν οι πράξεις αυτές.

Επίσης η συνάρτηση αυτή έχει και το όρισμα visualize το οποία αν είναι ίσο με 1 θα εμφανίσει σχετικά επεξηγηματικά διαγράμματα.

3.4 Υπολογισμός θεμελιώδους συχνότητας

Η συνάρτηση **pitchTracking** δέχεται σαν είσοδο ένα σήμα και υπολογίζει την θεμελιώδη συχνότητα. Όπως αναφέρθηκε και στην αλγοριθμική περιγραφή, η θεμελιώδης συχνότητα υπολογίζεται με την αυτοσυσχέτιση. Το Matlab παρέχει την συνάρτηση xcorr η οποία και χρησιμοποιείται για τον υπολογισμό της αυτοσυσχέτισης και μάλιστα με τον παράγοντα 'unbiased', για να χρησιμοποιεί τον αμερόληπτο τύπο και στη συνέχεια γίνεται κανονικοποίηση.

Επιπλέον η συνάρτηση **pitchTracking** εμφανίζει το διάγραμμα που φαίνεται στο σχήμα 19. Με κόκκινο εμφανίζονται τα k που είναι μέσα στο διάστημα $[k1, k2] = [13, 88]$. Το k με τη μεγαλύτερη αυτοσυσχέτιση μέσα σε αυτό το διάστημα είναι το 71, άρα η θεμελιώδης συχνότητα είναι $8000/71 = 112.67Hz$.



Σχήμα 19: Αυτοσυσχέτιση σήματος φωνής

3.5 Φασματική Ανάλυση

Η συνάρτηση **extractFeatures** χρησιμοποιείται για την εξαγωγή του διανύσματος χαρακτηριστικών του κάθε σήματοςμ με την διαδικασία που περιγράφεται στην αλγοριθμική περιγραφή. Επίσης σε αυτή η συνάρτηση αν δωθεί παράμετρος visualize=1 τότε θα εμφανίσει το φασματογράφημα του σήματος (όπως φαίνεται στο σχήμα 12).

3.6 Κυρίως πρόγραμμα

Το ASR σύστημα έχει υλοποιηθεί και σαν Matlab script (**main.m**) και σαν Matlab function (**numbersASR.m**). Το Matlab script **main.m** πραγματοποιεί τα εξής βήματα:

1. Ανοίγει ένα wav αρχείο φωνής
2. κανονικοποιεί τις τιμές του σήματος σε $[-1, 1]$
3. Αλλαγή ρυθμού δειγματοληψίας στα $8.000Hz$
4. Φιλτράλισμα με το Υλιλοπερατό FIR φίλτρο
5. εξαγωγή λέξεων μεσω της συνάρτησης **isolateWords**
6. για κάθε λέξη υπολογίζεται η θεμελιώδης συχνότητα με την συνάρτηση **pitchTracking**, το διάνυσμα χαρακτηριστικών με την συνάρτηση **extractFeatures**, το οποίο θα εισαχθεί σε ένα trained SVM Compact model το οποίο θα κάνει την ταξινόμηση.

7. Το script θα παίξει τις ανιχνευμένες λέξεις με περίπου 1 δευτερόλεπτο πάυσης ανάμεσα στη κάθε μία.

Επίσης το script θα εμφανίσει και ορισμένα επεξηγηματικά παράθυρα που θα δούμε στην επόμενη ενότητα.

Η συνάρτηση **numbersASR.m** είναι ουσιαστικά μία αντιγραφή του παραπάνω script με μόνη διαφορά ότι δεν "παίζει" στα ηχεία του υπολογιστή τις ανιχνευμένες λέξεις.

3.7 Μοντέλο SVM

Στον πηγαίο κώδικα υπάρχει ο φάκελος **trainedModelsCompact** και περιέχει δύο μοντέλα: το **svmClassifierV0.mat** και το **svmClassifierV1.mat**. Το πρώτο μοντέλο έχει εκπαιδευτεί με τις κλάσεις **zero**, **one**, **two**, **three**, **four**, **five** ενώ το δεύτερο περιέχει επιπλέον το **nine**. Στο **main** script και στην συνάρτηση **numbersASR.m** μπορεί να επισημανθεί ποιο μοντέλο θα χρησιμοποιηθεί.

Τα μοντέλα αυτά εκπαιδεύτηκαν μέσω scripts τα οποία περιλλαμβάντονται στα αρχεία του κώδικα. Αν θέλουμε να εκπαιδεύσουμε ένα μοντέλο πρέπει να τρέξουμε τα εξής scripts με την σειρά που δίνεται:

1. **preprocess_dataset_script.m** : Δημιουργία επαυξημένου συνόλου δεδομένων (Augmented Dataset)
2. **feature_extraction_script.m** : Εξαγωγή Διανυσμάτων χαρακτηριστικών
3. **SVM_training_script.m** : Εκπαίδευση μοντέλου
4. **SVM_evaluation_script.m** : Αξιολόγηση επίδοσης μοντέλου (περιλαμβάνει τις μετρικές που αναφέρθηκαν στην Αλγοριθμική Περιγραφή)

4 Οδηγίες Χρήσης - Παραδείγματα εκτέλεσης

Αρχικά αποσυμπιέζουμε τα source2023.zip και auxiliary2023.zip. Μέσα στον φάκελο source είναι όλα τα απαραίτητα .m files για το matlab. Αντιγράφουμε τον φάκελο samples, που βρίσκεται μέσα στο auxiliary2023, μέσα στο source φάκελο για να έχουμε πρόσβαση σε δοκιμαστικά αρχεία ήχου.

Τα αρχεία στο φάκελο samples είναι ηχογραφημένα από τον φοιτητή και περιέχονται και κάποια από την ιστοσελίδα του Rabiner.

Στον φάκελο auxiliary2023 υπάρχουν επιπλέον τα αρχεία **interpolation_example.m**, και **src_demo_script.m** τα οποία είναι απλά προσομοιωτικά scripts για χρήση συναρτήσεων. Επιπλέον το αρχείο **datasetParser.sh** περιέχει bash εντολές για την δημιουργεία dataset εκπαίδευσης.

Εκτελούμε το script main.m.

Το srcipt διάβασε το αρχείο 3rec.wav που περιέχει τη φωνή του φοιτητή και τις λέξεις 'one', 'two', 'three', 'four'.

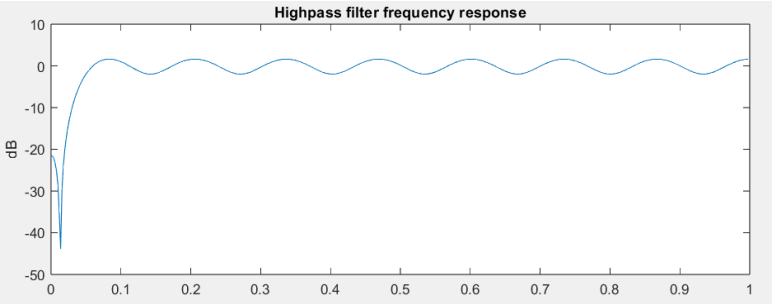
Το script εμφάνησε τα παρακάτω στο cmd:

Detected word#1: pitch = 109.589041, class = one
Detected word#2: pitch = 106.666667, class = two
Detected word#3: pitch = 108.108108, class = three
Detected word#4: pitch = 400.000000, class = four

Παραπάνω φαίνεται η θελελιώσης συχνότητα κάθε εντοπισμένης λέξης και η προβλεπόμενη κλάση. Βλέπουμε ότι το script αναγνώρισε σωστά τις λέξεις

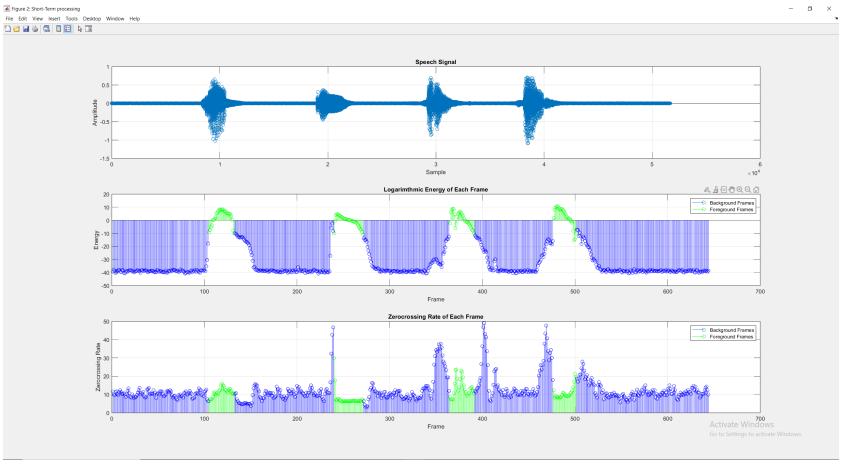
Το script επίσης έπαιξε στα ηχεία του υπολογιστή τις λέξεις του αρχείου με ενδίαμεσες παύσεις.

Επίσης εμφανίζονται τα παρακάτω παράθυρα:

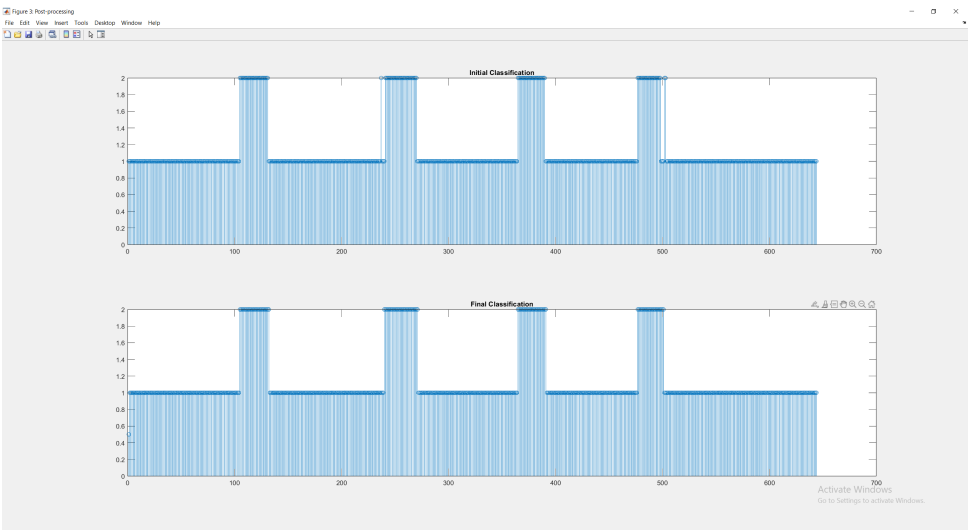


Σχήμα 20: Απόκριση συχνότητας του υψιλοπερατού φίλτρου

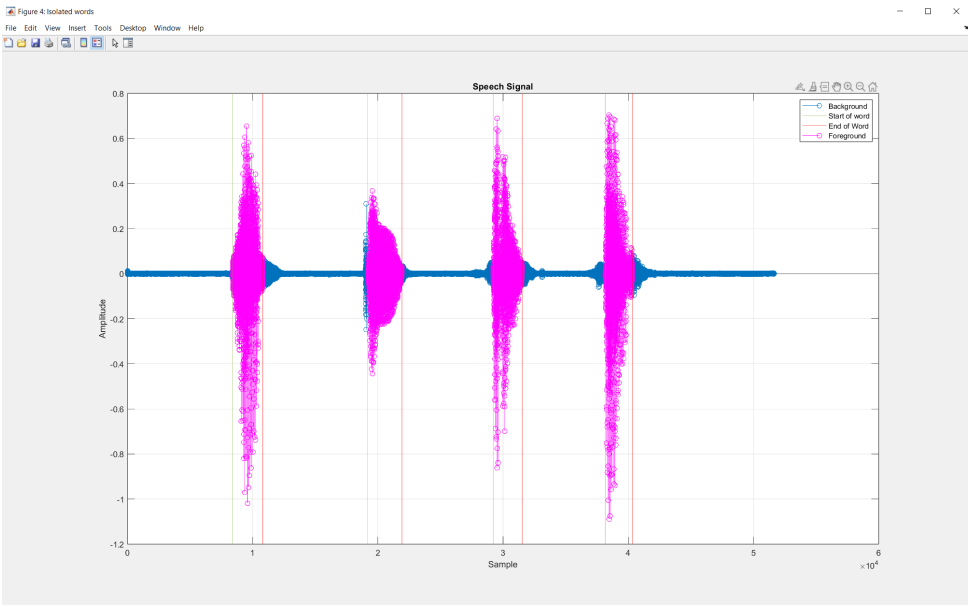
Τα επόμενα παράθυρα προέχρονται από την συνάρτηση **isolatedWords**



Σχήμα 21: Σήμα φωνής, Ενέργεια βραχέως χρόνου και Zero-crossing rate χωρίς το FIR φίλτρο

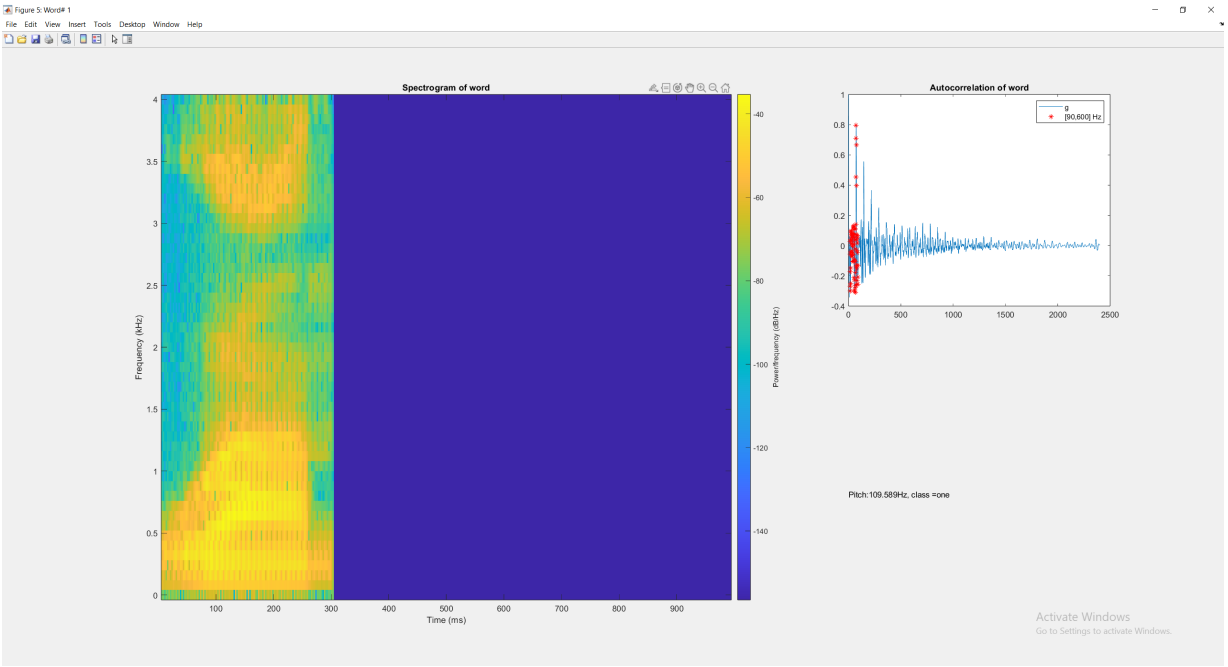


Σχήμα 22: Ταξινόμηση foreground vs background

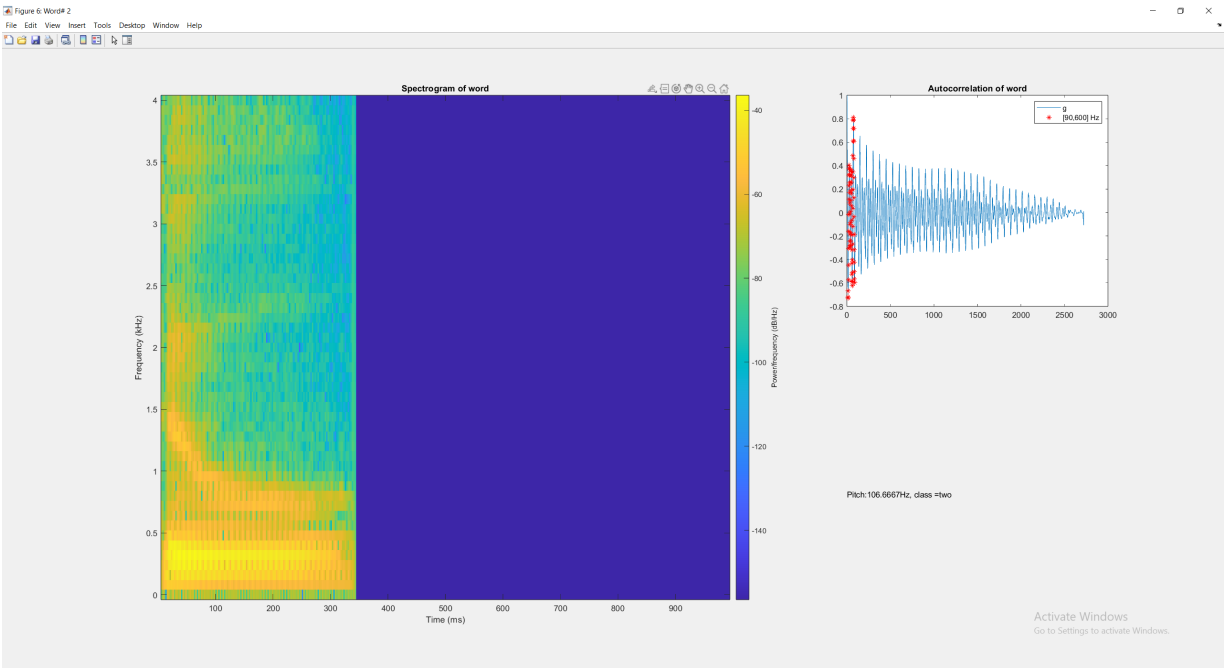


Σχήμα 23: Εξαγωγή λέξεων

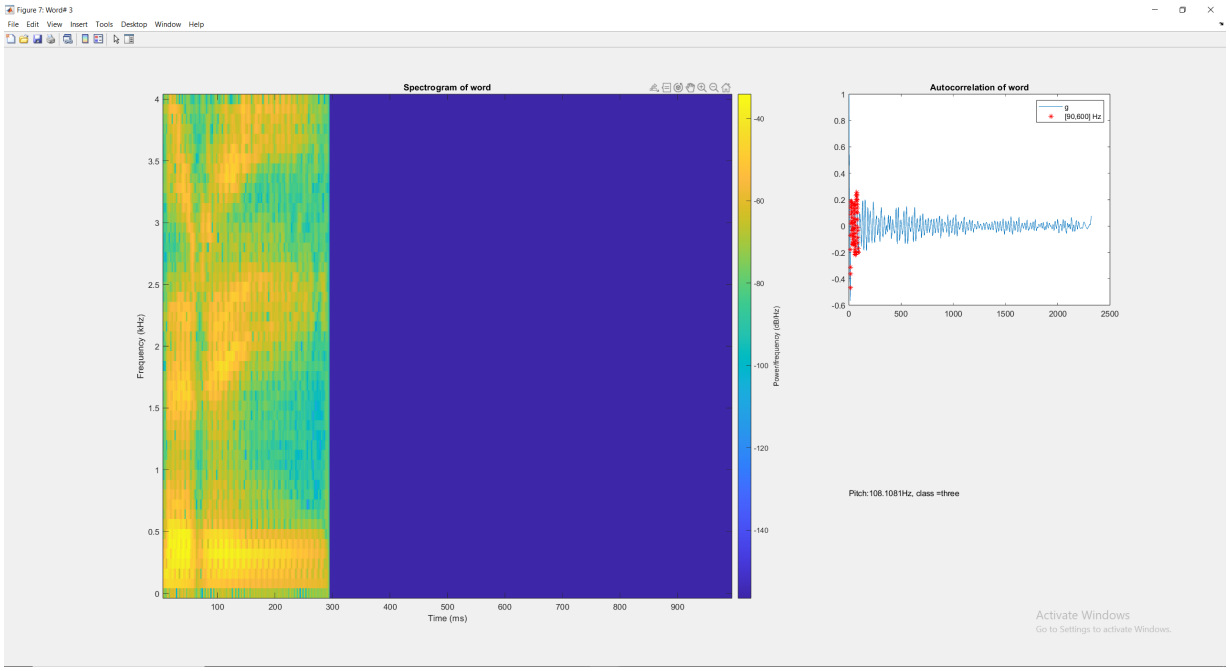
Στη συνέχεια εμφανίζεται το φασματογράφημα και η αυτοσυσχέτιση της κάθε λέξης.



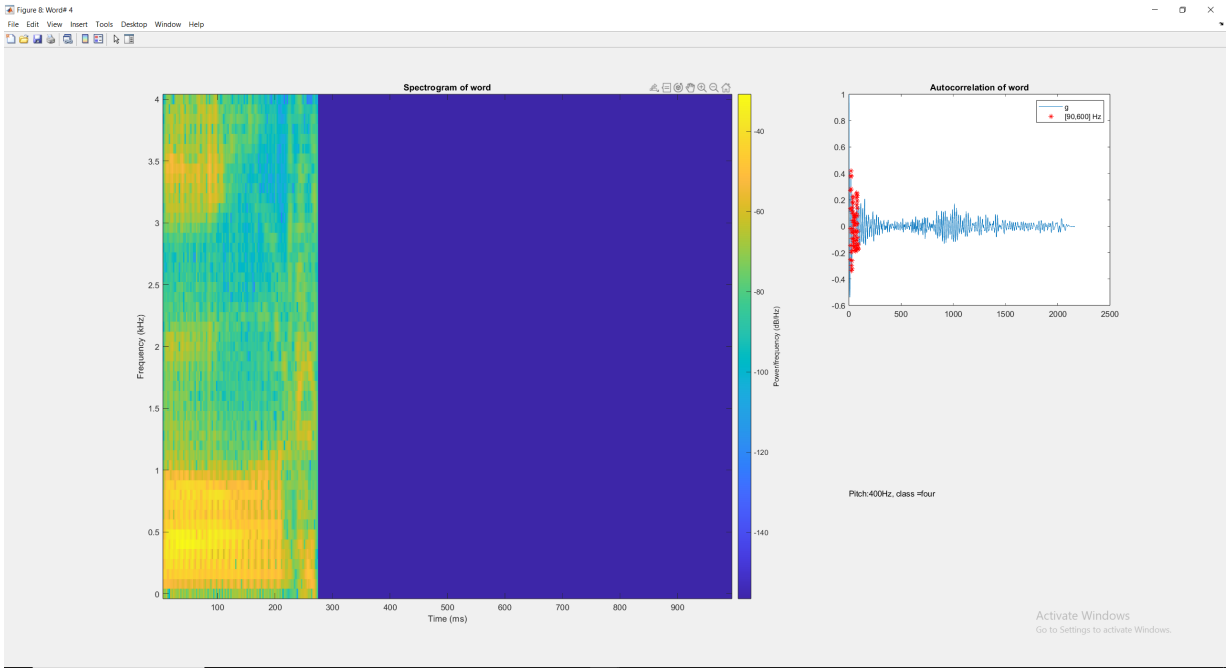
Σχήμα 24: Λέξη # 1



Σχήμα 25: Λέξη # 2



Σχήμα 26: Λέξη # 3



Σχήμα 27: Λέξη # 4

Μπορούμε να επαναλάβουμε την εκτέλεση του script με οποιοδήποτε αρχείο μέσα στο φάκελο samples.

Εναλλακτικά μπορούμε να χρησιμοποιήσουμε τη συνάρτηση **numbersASR**. Πατώντας στο Matlab cmd '*help numbersASR*' εμφανίζεται το εξής:

```
Example 1:

  load trainedModelsCompact\svmClassifierV0.mat;

  [speech,FsOrig]=audioread('samples/3rec.wav');

  [Y,pitch] = numbersASR(speech,FsOrig,SVMClassifierCompact,1)


Example 2:

  r = audiorecorder(44100,16,1);
  record(r) % speak into microphone
  stop(r)
  p = play(r); % listen

  speech = getaudiodata(r);

  load trainedModelsCompact\svmClassifierV0.mat;

  [Y,pitch] = numbersASR(speech,44100,SVMClassifierCompact,0)

fx >>
```

Σχήμα 28: Παραδείγματα χρήσης συνάρτησης numbersASR

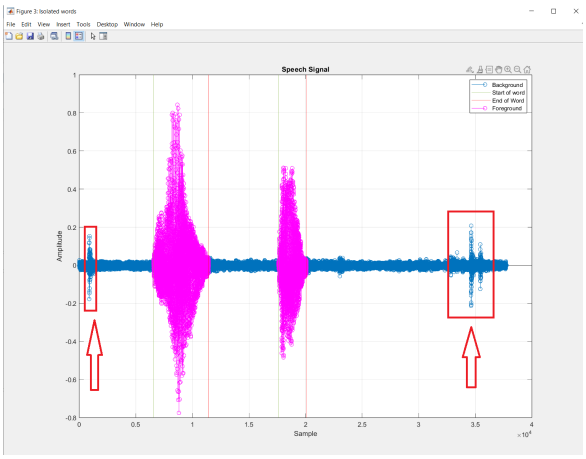
Αντιγράφουμε τις εντολές από το Example 2. Όταν κάνουμε record λέμε πχ τις λέξεις 'one' και 'two'. Η συνάρτηση επέστρεψε τα παρακάτω:

```
>> [Y,pitch] = numbersASR(speech,FsOrig,SVMClassifierCompact,0)

Y =

  1×2 string array

    "one"    "two"


pitch =

  115.9420  112.6761

fx >>
```

Σχήμα 29: Αποτελέσματα συνάρτησης numbersASR

Μπορούμε να τρέξουμε πάλι την συνάρτηση με όρισμα visualize==1, δηλαδή:
[Y,pitch] = numbersASR(speech,FsOrig,SVMClassifierCompact,1)

και θα εμφανιστούν τα διαγράμματα όπως και στο main script. Μάλιστα στο διάγραμμα του σχήματος 30 που φαίνονται οι εντοπισμένες λέξεις υπάρχουν στα άκρα (βλέπε κόκκινα βέλη) οι ήχοι που οφείλονται στο πληκτρολόγιο.



Σχήμα 30: Ηχογραφημένες λέξεις

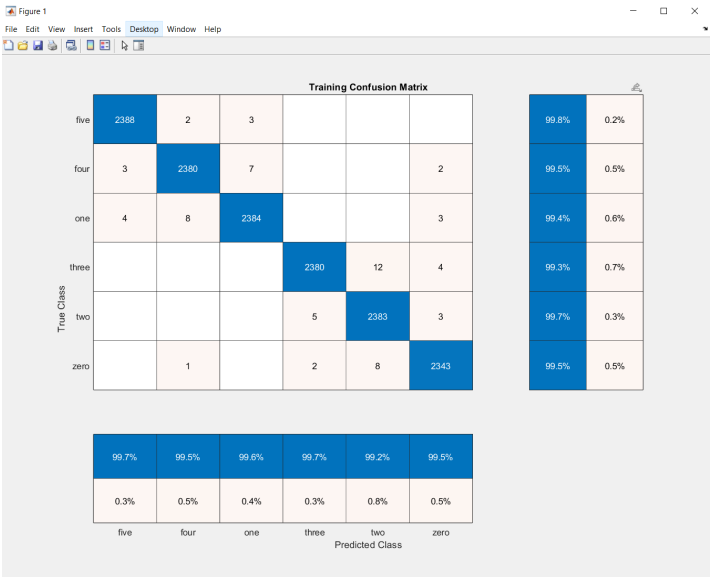
Προκειμένου να εκτελέσουμε το δικό μας μοντέλο πρέπει να εκτελέσουμε τα παρκάτω script με αυτή τη σειρά:

1. **preprocess_dataset_script.m** : Δημιουργία επαυξημένου συνόλου δεδομένων (Augmented Dataset)
2. **feature_extraction_script.m** : Εξαγωγή Διανυσμάτων χαρακτηριστικών
3. **SVM_training_script.m** : Εκπαίδευση μοντέλου
4. **SVM_evaluation_script.m** : Αξιολόγηση επίδοσης μοντέλου

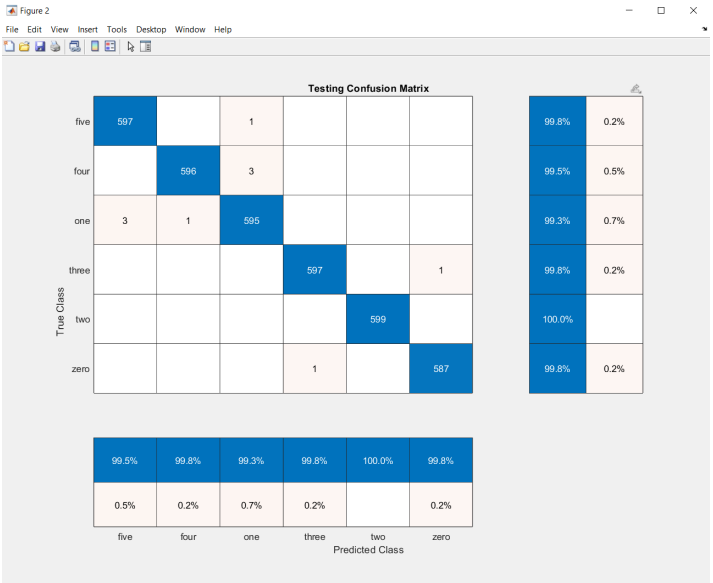
5 Συμπεράσματα και παρατηρήσεις

- Μοντέλο **svmClassifierV0**

  Το resubstitution error είναι ίσο με μηδέν και το 10-fold cross-validation error είναι ίσο με 0.01985. Αυτό σε γενικές γραμμές δείχνει ότι το μοντέλο έχει πετύχει μία καλή γενικότητα. Παρακάτω δίνονται οι πίνακες σύγχυσης για το training set και το testing set.



Σχήμα 31

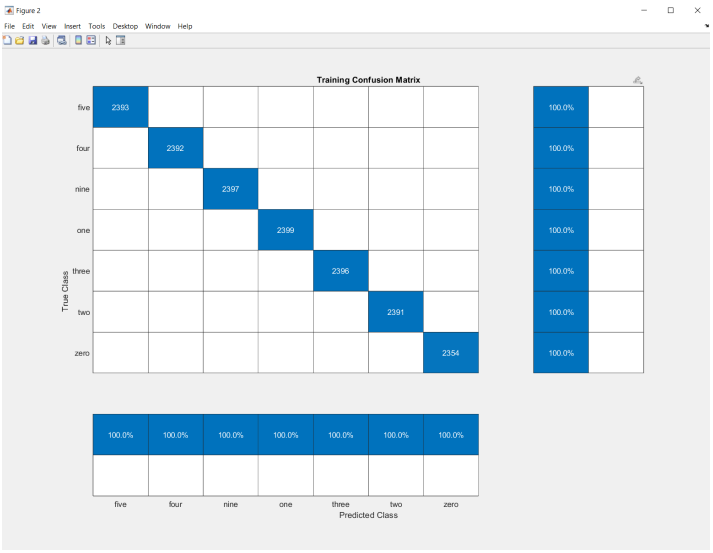


Σχήμα 32

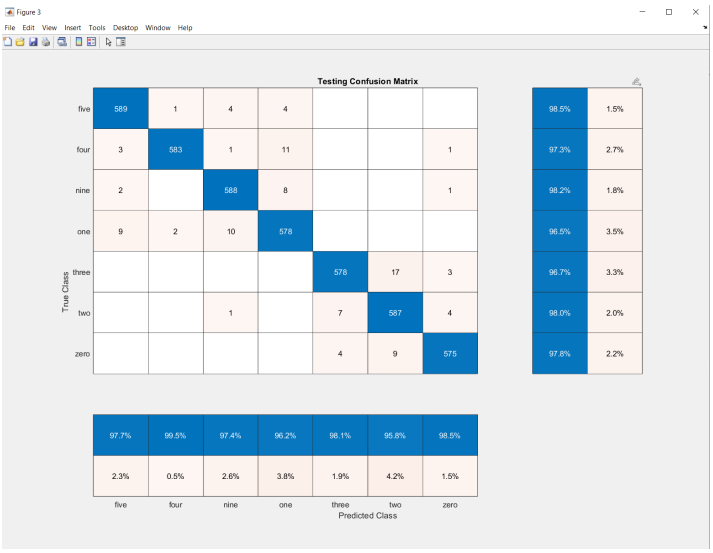
Οι τιμές είναι συγκεντρωμένες στη διαγώνιο, δηλαδή τα περισσότερα δείγματα ταξινομήθηκαν σωστά.

- Μοντέλο **svmClassifierV1**

  Παρόμοια με πριν, το resubstitution error είναι ίσο με μηδέν και το 10-fold cross-validation error είναι ίσο με 0.0234. Θεωρούμε πάλι πως το μοντέλο έχει πετύχει καλή γενικότητα. Παρακάτω δίνονται οι πίνακες σύγχυσης για το training set και το testing set.



Σχήμα 33



Σχήμα 34

Αντίστοιχα, οι τιμές είναι συγκεντρωμένες στη διαγώνιο, δηλαδή τα περισσότερα δείγματα ταξινομήθηκαν σωστά. Από τους πίνακες αυτούς μπορούν εύκολα να υπολογιστούν μέτρα όπως Precision, Recall και Accuracy.

Αναφορές

[1] S. Becker, M. Ackermann, S. Lapuschkin, K.-R. Müller, and W. Samek, "Interpreting and explaining deep neural networks for classification of audio signals," *CoRR*, vol. abs/1807.03418, 2018.

[2] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*. Prentice-hall Englewood Cliffs, second ed., 1999.

[3] R. G. Lyons, *Understanding Digital Signal Processing (2nd Edition)*. USA: Prentice Hall PTR, 2004.

[4] L. Rabiner and R. Schafer, *Theory and applications of digital speech processing*. Prentice Hall Press, 2010.