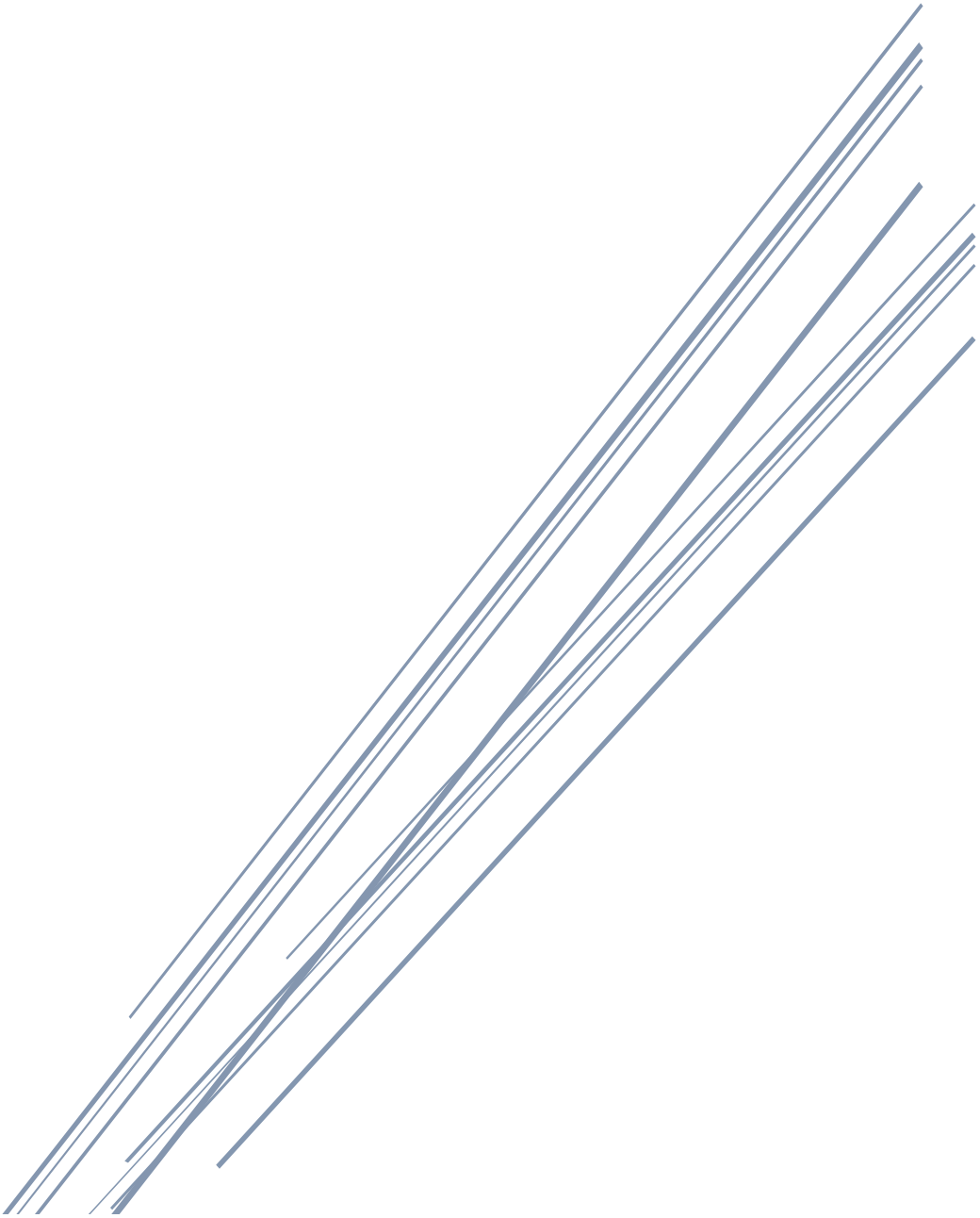
# Επεξεργασία Σημάτων Φωνής και Ήχου

## Απαλλακτική Εργασία

*Ακαδημαϊκό Έτος 2020 - 2021*




ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
UNIVERSITY OF PIRAEUS

Παναγιώτης Αποστολόπουλος, Π17007
Δημήτρης Ματσαγγάνης, Π17068
Παύλος Ρουμελιώτης, Π17112
Σκαρπέλος Αλέξανδρος, Π17122

# Περιεχόμενα

# 1. Εκφώνηση Εργασίας

Θέμα 1 (8 βαθμοί): 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.

1) Το σύστημα προχωρά στην κατάτμηση της πρότασης χρησιμοποιώντας υποχρεωτικά έναν ταξινομητή background vs foreground της επιλογής σας.

2) Στη συνέχεια αναγνωρίζει κάθε λέξη χρησιμοποιώντας ως φασματική αναπαράσταση μόνο το Mel-Spectrogram. Αν χρειαστείτε δεδομένα εκπαίδευσης, χρησιμοποιήστε μόνο σύνολο(α) δεδομένων από το site OpenSLR.

3) Στην έξοδο παράγεται κείμενο με τα ψηφία που αναγνωρίστηκαν.

• Δώστε έμφαση στην επεξεργασία του σήματος, προτού αρχίσουν τα στάδια κατάτμησης/αναγνώρισης (π.χ., με κατάλληλα φίλτρα, αλλαγή ρυθμού δειγματοληψίας, κ.λ.π).

• Είναι σημαντικό να περιγράψετε το σύστημα αλγοριθμικά (εξαγωγή χαρακτηριστικών, αλγόριθμος αναγνώρισης) και να εξηγήσετε τις επιδόσεις του χρησιμοποιώντας τις κατάλληλες μετρικές.

• Πρέπει να εξηγήσετε ποια δεδομένα χρησιμοποιήσατε κατά τον έλεγχο και την εκπαίδευση του συστήματος. Αν είναι δικά σας, πώς τα δημιουργήσατε.

• Προσπαθήστε να μην εξαρτάται το σύστημα από τα χαρακτηριστικά της φωνής του ομιλητή, αλλά να είναι όσο το δυνατόν ανεξάρτητο ομιλητή.

# 2. Εισαγωγή

Ζητούμενο της συγκεκριμένης εργασίας είναι η υλοποίηση ενός Automatic Speech Recognition συστήματος, το οποίο θα δέχεται μια ηχογράφηση μιας πρότασης αποτελούμενη από 5-10 ψηφία της Αγγλικής γλώσσας και το οποίο με τις κατάλληλες αλγοριθμικές διαδικασίες θα παράγει ένα κείμενο με τα ψηφία που λέχθηκαν.

Αρχικά, για το αρχείο εισόδου θα πρέπει να δοθεί ιδιαίτερη έμφαση στην επεξεργασία του σήματος, πριν την διαδικασία κατάτμησης και αναγνώρισης, χρησιμοποιώντας κατάλληλα φίλτρα αλλά και αλλάζοντας τον ρυθμό δειγματοληψίας. Με την χρήση ενός *FIR Pass φίλτρου* επιτυγχάνεται η ποιοτική βελτίωση του σήματος, ενώ με την αλλαγή του ρυθμού δειγματοληψίας μπορούμε να μειώσουμε τον αριθμό των δειγμάτων ανά δευτερόλεπτο, καθιστώντας τους υπολογισμούς πιο γρήγορους.

Στη συνέχεια, με την βοήθεια ενός ταξινομητή **Background vs Foreground**, ταξινομούνται στο background οτιδήποτε δεν είναι σήμα ομιλίας και στο foreground το σήμα ομιλίας με τη βοήθεια του short term processing, και την εύρεση του *Root Mean Square Energy (RMSE)* και του *Zero Crossing Rate (ZCR)* για κάθε παράθυρο. Πιο αναλυτικά, θα σπάει το σήμα σε κομμάτια (windows), τα οποία θα είναι μη επικαλυπτόμενα, δηλαδή το *HOP_LENGTH = FRAME_ LENGTH,* και για κάθε κομμάτι σήματος θα παίρνει μια απόφαση κατά πόσο είναι σήμα ομιλίας ή σήμα υποβάθρου (background), αξιοποιώντας τα παραπάνω χαρακτηριστικά.

Έπειτα, κάθε λέξη θα αναγνωρίζεται χρησιμοποιώντας τη φασματική αναπαράσταση μόνο του *Mel-Spectrogram*. Αυτό θα αξιοποιηθεί καταλλήλως, προκειμένου να εξαχθούν τα *MFCC features*, τα οποία ταξινομούνται στο **Random Forest Classifier**. Για τη δημιουργία του ταξινομητή χρησιμοποιήθηκαν δεδομένα, τα οποία κατασκευάσαμε εμείς οι ίδιοι. Επόμενο στάδιο, είναι η πρόβλεψη των ψηφίων του εισαγόμενου σήματος. Τέλος, προβάλλονται τα βοηθητικά διαγράμματα και χρησιμοποιούνται μετρικές για το ποσοστό επιτυχία των ταξινομητών.

Σαν υποθέσεις εργασίας κάναμε τις εξής:

1. Για τη δημιουργία του Dataset κάθε ομιλητής ηχογράφησε όλους τους αριθμούς από δέκα φορές τον έκαστο, χρησιμοποιώντας το λογισμικό *Audacity*, μεριμνώντας να περιορίσουμε όσο το δυνατόν περισσότερο το θόρυβο.
2. Το FRAME_LENGTH είναι ίσο με το HOP_LENGTH και ίσο με 256 δείγματα ή 32 ms, δηλαδή δεν υπάρχει επικάλυψη
3. Ο ρυθμός δειγματοληψίας καθορίστηκε στα 8 kHz.
4. Για το αρχείο εισόδου για το οποίο καλούμαστε να εφαρμόσουμε την παραπάνω διαδικασία, επιλέχθηκε η χρήση κενού μεταξύ των ψηφίων, διαστήματος περίπου 500 ms.

# 3. Υλοποίηση

Για την υλοποίηση και την επίλυση της άσκησης χρησιμοποιήθηκε η γλώσσα προγραμματισμού *Python v.3.7.9*, οι απαιτήσεις για την εκτέλεση παρατίθενται στην **Ενότητα 6**. Τα βήματα που ακολουθήθηκαν για την υλοποίηση του συστήματος θα αναλυθούν διεξοδικά παρακάτω.

## 3.1 Προεπεξεργασία

Στη συγκεκριμένη ενότητα θα αναφερθούμε στις απαραίτητες ενέργειες που γίνονται από την εφαρμογή μας με σκοπό την βέλτιστη επεξεργασία του, τόσο του συνόλου εκπαίδευσης, όσο και του εισαγόμενου αρχείου προς αναγνώριση.

### *3.1.1 Εισαγωγή αρχείων & Ρυθμός δειγματοληψίας*

Αρχικά, για κάθε αρχείο που βρίσκεται στο σύνολο εκπαίδευσης (***dataset***), ορίζουμε τον ρυθμό δειγματοληψίας, *sample rate*, ίσο με 8 kHz, αφού μετά από έρευνα, αυτός ο ρυθμός επιτρέπει την καλύτερη καταγραφή και ποιοτικότερη επεξεργασία του σήματος.

Ομοίως και για το σήμα εισόδου, ορίζουμε τον ρυθμό δειγματοληψίας ίσο με 8 kHz, ενώ για τον σκοπό αυτό υλοποιήθηκε μια συνάρτηση, η ***preprocessing***.

### *3.1.2 FIR Band Pass Filter*

Το πρώτο βήμα για την επεξεργασία τόσο των αρχείων που βρίσκονται στο σύνολο εκπαίδευσης, όσο και του ηχογραφημένου σήματος, είναι η διαδικασία φιλτραρίσματος μέσω ενός FIR filter band (*Finite Impulse Response*). Ένα band pass filter, είναι μία διαδικασία μέσω της οποίας γίνονται δεκτές οι συχνότητες ενός συγκεκριμένου πεδίου ορισμού, ενώ ταυτόχρονα απορρίπτονται οι συχνότητες εκτός αυτού του πεδίου.

Στην υλοποίησή μας χρησιμοποιούμε ένα δικό μας band pass filter, το οποίο όπως προαναφέρθηκε χαρακτηρίζει ως αποδεκτά τα φίλτρα μεταξύ δύο συχνοτήτων *fL (frequency Low)* και *fH (frequency High)* και

απορρίπτει τις συχνότητες εκτός του παραπάνω πεδίου. Οι πλέον κατάλληλες συχνότητες για το βέλτιστο φιλτράρισμα είναι τα 200 Hz για το *fL* και τα 4000 Hz για το *fH.*

Στη συνέχεια, υπολογίζουμε τα άνω και κάτω φράγματα με βάση τις συχνότητες *fL* και *fH,* συνελίσσουμε τα δύο αποτελέσματα και τα εφαρμόζουμε στο εισερχόμενο σήμα. Η παραπάνω διαδικασία υλοποιείται με τη βοήθεια της βιβλιοθήκης ***numpy*** και των μεθόδων που αυτή μας παρέχει.

## 3.2 Dataset

Στον φάκελο *training* υπάρχουν ξεχωριστοί φάκελοι για κάθε ψηφίο από το 0 έως το 9, όπου ο καθ΄ ένας από αυτούς περιέχει 10 ηχογραφημένα παραδείγματα από τον κάθε ομιλητή. Το πρόγραμμα εκτελεί αναζήτηση για το κάθε αρχείο και το ταυτίζει με το κάθε ψηφίο. Μετά το αρχείο υπόκειται στην προεπεξεργασία, όπως αυτή περιεγράφηκε στην παραπάνω ενότητα και έπειτα γίνεται η εξαγωγή των *MFCC* χαρακτηριστικών, που θα εξηγηθεί εκτενώς στην Ενότητα 3.6***.*** Αφού ολοκληρωθεί η εξαγωγή αυτών, με τη βοήθεια της βιβλιοθήκης ***numpy*** υπολογίζουμε την μέση τιμή αυτών χρησιμοποιώντας την σχετική μέθοδο.

Ακολούθως, τα συμπεράσματα που προέκυψαν για το κάθε αρχείο, καθώς και τα ψηφία που αντιπροσωπεύουν εισάγονται σε ένα *Random Forest Classifier*. Ο συγκεκριμένος ταξινομητής υλοποιείται με τη βοήθεια της βιβλιοθήκης ***sklearn*** και για τον σκοπό αυτό δημιουργήθηκε η συνάρτηση ***build_dataset***, η οποία επιστρέφει τον ταξινομητή.

## 3.3 Ενέργεια Σήματος

Σε αυτή την ενότητα θα αναφερθούμε στην ενέργεια του σήματος και τη χρησιμότητα της στην υλοποίησή μας.

Αρχικά, αξίζει να αναφερθούμε στον ορισμό της ενέργειας του σήματος. Πιο συγκεκριμένα, ως ενέργεια σήματος ορίζουμε το άθροισμα των τετραγώνων των φωνημάτων (των κυματομορφών - magnitudes).

Ωστόσο, ως παράμετρο της εφαρμογής μας δεν χρησιμοποιούμε την ενέργεια, αλλά υπολογίζουμε τη μέση τετραγωνική ρίζα αυτής (*Root Mean Square Energy – RMSE*) για κάθε παράθυρο (frame) των ηχητικών σημάτων. Αυτό γίνεται με τη βοήθεια της βιβλιοθήκης της Python, ***librosa*** και συγκεκριμένα τη μέθοδο *librosa.feature.rms*, η οποία υπολογίζει με αρκετά γρήγορο τρόπο τη μέση τετραγωνική ρίζα της ενέργειας, καθώς δεν απαιτεί τον υπολογισμό του STFT (*Short-Time Fourier Transform*).

Συνεπώς, ορίζοντας τις παραμέτρους που απαιτεί η μέθοδος (το ηχητικό σήμα, το μήκος παραθύρου και το μέγεθος του βήματος), συνδυαστικά με την χρήση της ***numpy*** λαμβάνουμε ως αποτέλεσμα έναν πίνακα που εμπεριέχει την ενέργεια των ακουστικών σημάτων. Αυτός ο πίνακας παίζει καθοριστικό ρόλο στην εύρεση των ουσιώδη φωνημάτων (*thresholds*) μέσα στο κάθε ηχητικό αρχείο. Προκειμένου να εξάγουμε την ενέργεια του σήματος δημιουργήσαμε τη συνάρτηση ***rmse***.

## 3.4 Zero Crossing Rate

Σημαντικό βήμα για την υλοποίηση του ταξινομητή *Background vs Foreground* είναι η εύρεση του ρυθμού διέλευσης από το μηδέν για το αρχείο εισόδου. Ο ρυθμός αυτός προκύπτει βρίσκοντας τον αριθμό των φορών που διέρχεται ένα σήμα του μηδενός από αρνητικές σε θετικές τιμές και αντιστρόφως διαιρώντας το με το μήκος του παραθύρου.

Η σημασία του εν λόγω ρυθμού έγκειται στο γεγονός ότι αξιοποιείται προκειμένου να μπορέσουμε να έχουμε μια άποψη για τη θορυβώδη ή όχι φύση του ήχου κάθε στιγμή. Όταν έχουμε ένα σήμα, στο οποίο υπάρχουν τμήματα, τα οποία οφείλονται στο σήμα του περιβάλλοντος υποθέτουμε ότι σε εκείνα τα σημεία το Zero Crossing Rate (ZCR) είναι μεγαλύτερο και η ενέργεια μικρότερη. Συνεπώς, ο συνδυασμός των δύο

αυτών παρατηρήσεων μας επιτρέπει να κόψουμε το χρήσιμο σήμα και να αποκόψουμε τα τμήματα του σήματος που οφείλονται στο περιβάλλον. Για την εύρεση του ZCR υλοποιήθηκε η συνάρτηση ***zero_crossing_rate***, στης οποίας το σώμα αξιοποιήθηκε η μέθοδος της βιβλιοθήκης ***librosa***, *librosa.feauture.zero_crossing_rate*, με ορίσματα το σήμα εισόδου, το μήκος του παραθύρου και το *hop_size*.

## 3.5 Background vs Foreground Classifier – Κατάτμηση Σήματος

Στη παρούσα ενότητα θα αναφερθούμε στον ταξινομητή *Background vs Foreground*, ο οποίος αξιοποιεί τόσο την ενέργεια του σήματος, όσο και το ρυθμό διέλευσης από το μηδέν (*Zero Crossing Rate*).

Πιο αναλυτικά, με τη βοήθεια της συνάρτησης ***b_vs_f***, θα διακρίνουμε τα φωνήματα των ψηφίων από τον ήχο του περιβάλλοντος. Σαν κατώφλι (threshold) της ενέργειας θεωρούμε την μέση τιμή αυτής, ενώ σαν κατώφλι του *ZCR* την μέση αυτού. Για την εύρεση ενός φωνήματος ελέγχουμε εάν το *Zero Crossing Rate* του εκάστοτε παραθύρου είναι χαμηλότερο ή ίσο από το αντίστοιχο κατώφλι και ταυτόχρονα η ενέργεια αυτού υψηλότερη ή ίση του κατωφλιού.

Το επόμενο βήμα, είναι η κατάτμηση του σήματος αξιοποιώντας τα αποτελέσματα της προηγούμενης διαδικασίας. Ειδικότερα, μετά από μια σειρά ελέγχων επιστρέφεται μια λίστα με τα φωνήματα για τα ψηφία που αναγνωρίστηκαν. Οι έλεγχοι αυτοί χρησιμοποιούνται για την εύρεση της αρχής και του τέλους ενός φωνήματος, αλλά και την απαλοιφή τυχών κενών που υπάρχουν εντός των παραθύρων που αναγνωρίστηκαν τα φωνήματα.

## 3.6 Εξαγωγή MFCC Χαρακτηριστικών

Το επόμενο στάδιο στην υλοποίησή μας είναι η εξαγωγή *των MFCC (Mel-Frequency Cepstral Coefficients)* χαρακτηριστικών, τα οποία όπως θα δούμε και στη συνέχεια θα χρησιμοποιηθούν για την εκπαίδευση του μοντέλου αναγνώρισης των ψηφίων.

Η τεχνική εξαγωγής χαρακτηριστικών *MFCC* περιλαμβάνει τη κατάτμηση του σήματος σε παράθυρα, την εφαρμογή του *DFT* (Discrete Fourier Transform), την αντιστοίχιση των συχνοτήτων στην *κλίμακα Mel*, αλλά

και την εφαρμογή του *αντίστροφου DCT*. Η λεπτομερής περιγραφή των επιμέρους βημάτων που εμπλέκονται στην εξαγωγή χαρακτηριστικών *MFCC* εξηγείται παρακάτω:

- Frame blocking and windowing:

  Το σήμα ομιλίας είναι ένα αργό στο χρόνο ή σχεδόν στατικό σήμα. Για σταθερά ακουστικά χαρακτηριστικά, η ομιλία πρέπει να εξεταστεί για αρκετά σύντομο χρονικό διάστημα.

  Επομένως, η ανάλυση ομιλίας πρέπει να πραγματοποιείται πάντα σε μικρά τμήματα στα οποία θεωρείται το σήμα ομιλίας στάσιμο. Ο σκοπός της αλληλεπικαλυπτόμενης ανάλυσης είναι ότι κάθε ήχος ομιλίας της ακολουθίας εισόδου θα είναι περίπου στο κέντρο κάθε frame.

- DFT spectrum:

  Εφαρμόζοντας τον DFT μετατρέπεται σε φάσμα μεγέθους κάθε windowed frame.

- Mel spectrum:

  Η αναπαράσταση του βραχυπρόθεσμου φάσματος ισχύος ενός ήχου, που βασίζεται σε έναν γραμμικό μετασχηματισμό ενός λογαριθμικού φάσματος ισχύος αναπαριστώμενο σε μια μη γραμμική *κλίμακα Mel*.

- Discrete Cosine Transform (DCT):

  Το DCT εφαρμόζεται στους μετασχηματισμένους συντελεστές συχνότητας Mel παράγοντας ένα σύνολο συντελεστών *cepstral*.

  Ως εκ τούτου, οι κορυφές των χαμηλών συχνοτήτων αντιπροσωπεύονται από ένα σήμα στον *cepstral* τομέα με την αντίστοιχη κορυφή αιχμής στο βήμα του σήματος και από έναν αριθμό παραγόντων.

Προκειμένου να εξαχθούν τα συγκεκριμένα χαρακτηριστικά, αξιοποιήθηκε η μέθοδος της βιβλιοθήκης ***librosa****, librosa.feauture.mfcc* .

Για το σύνολο εκπαίδευσης η μέθοδος δέχεται σαν όρισμα το εκάστοτε αρχείο, τον ρυθμό δειγματοληψίας, το *FRAME_SIZE* και το *HOP_SIZE*.

Αντίστοιχα, για το αρχείο εισόδου τα ορίσματα είναι τα φωνήματα που έχουν προκύψει από την κατάτμηση, αλλά και ο ρυθμός δειγματοληψίας, το *FRAME_SIZE* και το *HOP_SIZE*. Έχοντας εξάγει τα χαρακτηριστικά αυτά, υπολογίζουμε την μέση τιμή αυτών χρησιμοποιώντας την σχετική μέθοδο της βιβλιοθήκης ***numpy***.

## 3.7 Αναγνώριση ψηφίων

Τελευταίο βήμα στην εν λόγω υλοποίηση, είναι η αξιοποίηση της μέσης τιμής των *MFCC* χαρακτηριστικών από τον *Random Forest Classifier*, ώστε να γίνει η πρόβλεψη των ψηφίων με βάση το εκπαιδευμένο μοντέλο, όπως αυτό έχει προκύψει από το σύνολο εκπαίδευσης. Με τη συνάρτηση ***recognition*** ζητάμε από το μοντέλο να προβλέψει το ψηφίο που αντιστοιχεί στα *MFCC* χαρακτηριστικά του κάθε φωνήματος, όπως αυτά είχαν προκύψει από τη διαδικασία της κατάτμησης και εκτυπώνεται το αποτέλεσμα.

Ένας *Random Forest Classifier* δημιουργεί πολλαπλά δέντρα απόφασης, τα οποία συγχωνεύονται προκειμένου να λάβουμε μια πιο ακριβή και εύρωστη πρόβλεψη, λόγω του μεγάλου πλήθους δέντρων που χρησιμοποιούνται. Ουσιαστικά δημιουργούνται δέντρα απόφασης με δεδομένα, τα οποία συλλέγονται από τυχαία δείγματα και στη συνέχεια λαμβάνουμε μια πρόβλεψη για τα επιμέρους δέντρα, ενώ η καλύτερη απόφαση επιλέγεται με τη διαδικασία της ψηφοφορίας αυτών.

# 4. Αποτελέσματα

Στη συγκεκριμένη ενότητα θα παρουσιαστούν τα αποτελέσματα από την εκτέλεση του προγράμματος.

## 4.1 Μετρικές

Οι μετρικές διαδραματίζουν σημαντικό ρόλο στην παρούσα εφαρμογή, καθώς παρέχουν τα ποσοστά επιτυχίας των ταξινομητών.

### *4.1.1 Μετρική Background vs Foreground Classifier*

Η μετρική του ταξινομητή *Background vs Foreground* χρησιμοποιεί όλα τα αρχεία του φακέλου *testing* και βρίσκει το ποσοστό επιτυχίας του. Ως επιτυχία ορίζεται η εύρεση του σωστού αριθμού ψηφίων για το εκάστοτε αρχείο.

### *4.1.2 Μετρική Random Forest Classifier*

Η μετρική του ταξινομητή *Random Forest* χρησιμοποιεί όλα τα αρχεία του φακέλου *testing*, για τα οποία βρέθηκε με επιτυχία το πλήθος των ψηφίων τους. Για το κάθε αρχείο που πληρεί τις προϋποθέσεις, προβλέπει μέσω του *Random Forest Classifier* (*βλ. Ενότητα 3.7*) τα ψηφία.

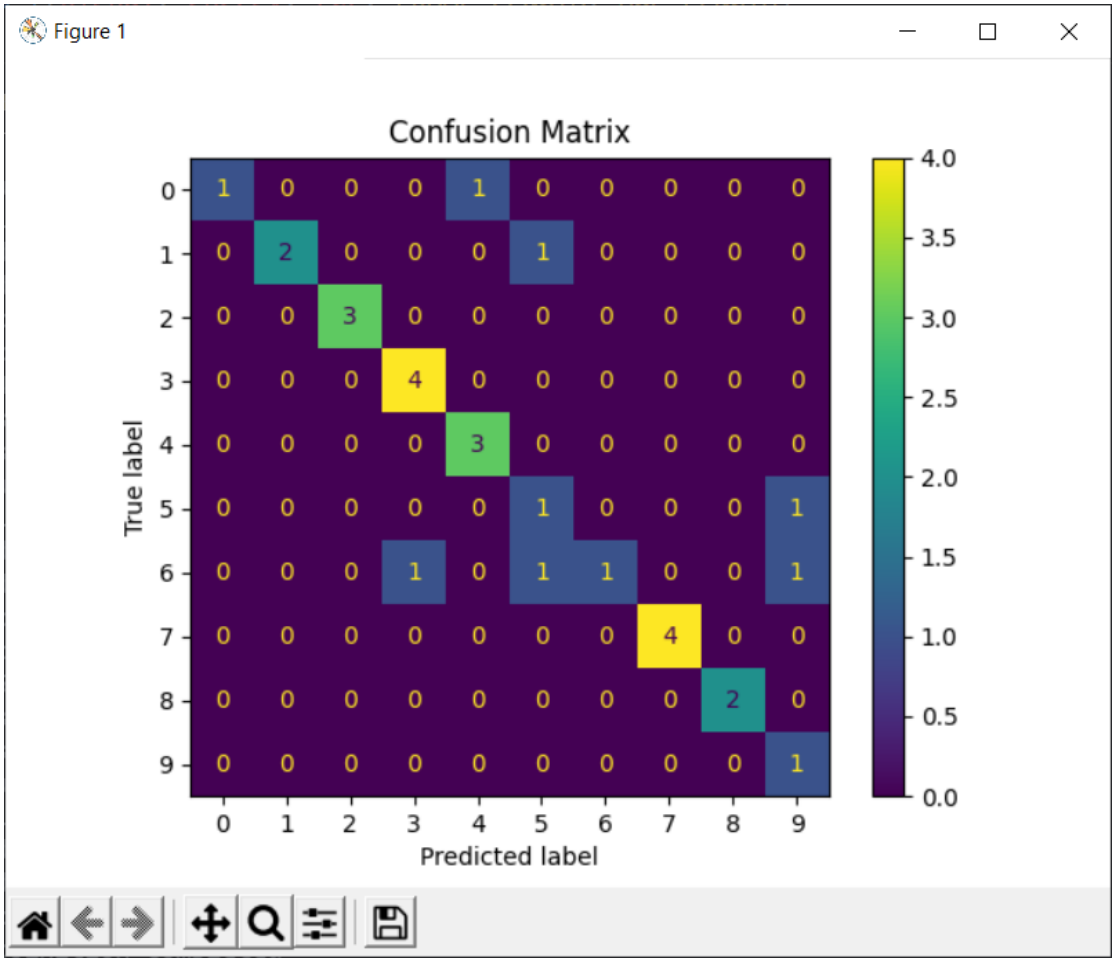
Έπειτα, υπολογίζει το ποσοστό επιτυχίας του εν λόγο ταξινομητή, όπου ως επιτυχία ορίζεται η εύρεση του σωστού ψηφίου.

Τέλος, αξίζει να αναφερθεί ότι στην παρούσα μετρική γίνεται χρήση ενός *Confusion Matrix*, ο οποίος περιγράφει την απόδοση του συγκεκριμένου μοντέλου ταξινόμησης σε ένα σύνολο δοκιμαστικών αρχείων (dataset), όπου συγκρίνονται οι πραγματικές με τις προβλεπόμενες τιμές.

Ακολουθούν εικόνες μετά από την κλήση της συνάρτησης για της μετρικές (*συνάρτηση* ***accuracy***):

```
[Result]: Background vs Foreground Classifier Accuracy: 83.33%
[Result]: Random Forest Classifier Accuracy: 78.57%
```

*Εικόνα 1: Αποτελέσματα ακρίβειας ταξινομητών*




*Εικόνα 2: Confusion matrix*

## 4.2 Διαγράμματα

Τα διαγράμματα αποτελούν σημαντικό κομμάτι της οπτικοποίησης της εργασίας μας και δεν θα μπορούσαμε παρά να μην αναφερθούμε σε αυτά εκτενώς.

Αξίζει να σημειωθεί το γεγονός ότι τα διαγράμματα Κυματομορφής – Ενέργειας

### *4.2.1 Διάγραμμα Κυματομορφής (Waveplot)*

Με το διάγραμμα κυματομορφής - waveplot μπορούμε να δούμε την κυματορφή του εισαγόμενου σήματος, καθώς και τη διάρκεια αυτού.

Στην πράξη όμως η υλοποίηση γίνεται με την βοήθεια της βιβλιοθήκης της ***librosa*** και συγκεκριμένα της μεθόδου *librosa.display.waveplot,* η οποία σχεδιάζει τη κυματομορφή του σήματος.

### *4.2.2 Διάγραμμα Ενέργειας (Root Mean Square Energy)*

Το διάγραμμα της μέσης τετραγωνικής ρίζας της ενέργειας μας παρέχει χρήσιμες πληροφορίες, οι οποίες αξιοποιούνται από τον ταξινομητή Background vs Foreground (*βλ. Ενότητα 3.5*) για την κατάτμηση του σήματος.

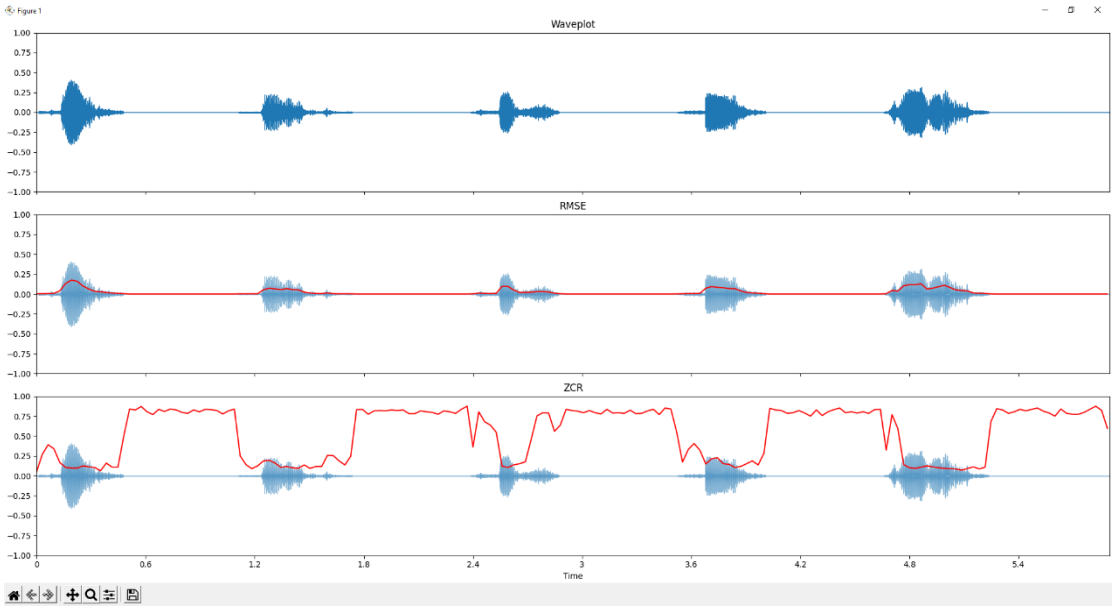
Η υλοποίηση του γίνεται με την βοήθεια της βιβλιοθήκης της ***librosa*** και συγκεκριμένα της μεθόδου *librosa.display.waveplot,* για την απεικόνιση της κυματομορφής του σήματος καθώς και της μεθόδου *librosa.frames_to_time*, για την για τη μετατροπή τους σε μονάδες χρόνου (δευτερόλεπτα).

### *4.2.3 Διάγραμμα Zero Crossing Rate*

Έπειτα, από το διάγραμμα του Zero Crossing Rate οπτικοποιούμε τη διαδικασία της εύρεσης του ρυθμού διέλευσης από το μηδέν και εξάγουμε χρήσιμες πληροφορίες, οι οποίες συνδυαστικά με την μέση τετραγωνική ρίζας της ενέργειας μας βοηθάνε στην εύρεση των φωνημάτων σε ένα ηχητικό σήμα.

Κατά αντιστοιχία με το αμέσως παραπάνω διάγραμμα, η υλοποίηση του παρόντος γίνεται με την βοήθεια της βιβλιοθήκης της ***librosa*** και συγκεκριμένα της μεθόδου *librosa.display.waveplot*, για την απεικόνιση της κυματομορφής του σήματος, καθώς και της μεθόδου *librosa.frames_to_time*, για τη μετατροπή τους σε μονάδες χρόνου (δευτερόλεπτα).

Ακολουθεί μία ενδεικτική εικόνα από τα τρία παραπάνω διαγράμματα:




*Εικόνα 3: Διαγράμματα Waveplot - RMSE – ZCR*

### *4.2.4 Διάγραμμα Spectrogram*

Με το διάγραμμα του *Spectrogram* αναπαριστούμε οπτικά το φάσμα του σήματος, όπως αυτό διακυμαίνεται στην πάροδο του χρόνου. Για να επιτύχουμε τη συγκεκριμένη οπτικοποίηση υπολογίζουμε διάφορα φάσματα εφαρμόζοντας τον *Fast Fourier Transform (FFT)* σε πολλαπλά τμήματα του σήματος, χωρισμένα σε παράθυρα. Η διαδικασία αυτή ορίζεται ως *Short-Time Fourier Transform (STFT)*. Αυτά τα πολλαπλά *FFTs* στοιβάζονται το ένα πάνω στο άλλο, απεικονίζοντας έτσι το εύρος της έντασης του σήματος με την πάροδο του χρόνου σε διαφορετικές συχνότητες. Επιπρόσθετα, ο κάθετος άξονας (y-axis) μετατρέπεται στη λογαριθμική κλίμακα (log scale) και τα απεικονιζόμενα χρώματα σε decibels (dB).

### *4.2.5 Διάγραμμα Mel-Spectrogram*

Το διάγραμμα *Mel-Spectrogram* χρησιμοποιείται με τρόπο παρόμοιο με αυτό του *Spectrogram*. Η ειδοποιός διαφορά των δύο διαγραμμάτων είναι ότι στο παρόν διάγραμμα η συχνότητα μετατρέπεται στην *Mel κλίμακα.*

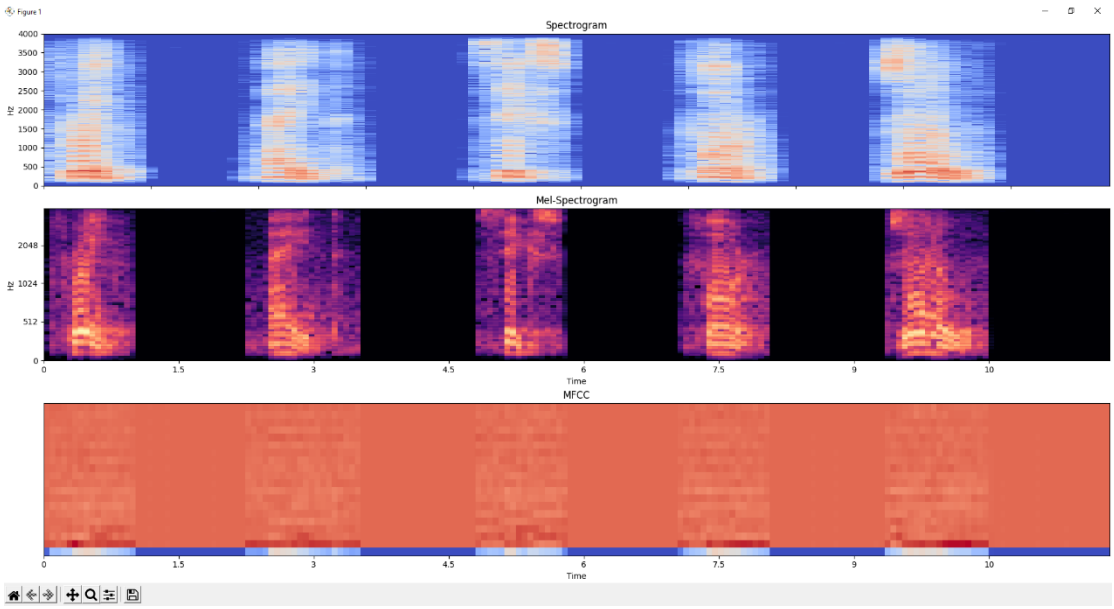
Είναι γενικώς γνωστό ότι οι άνθρωποι δεν αντιλαμβανόμαστε τις συχνότητες σε μία γραμμική κλίμακα. Ειδικότερα, είμαστε καλύτεροι στο να καταλαβαίνουμε τις διαφορές σε χαμηλότερες παρά σε υψηλότερες συχνότητες. Ως εκ τούτου, η μετατροπή στη κλίμακα Mel είναι σημαντική για την υλοποίησή μας, καθώς είναι το αποτέλεσμα ενός μη γραμμικού μετασχηματισμού της κλίμακας της συχνότητας. Αυτή η κλίμακα κατασκευάζεται με τέτοιο τρόπο, ώστε οι ήχοι που έχουν ίση απόσταση μεταξύ τους στη κλίμακα Mel, θα έχουν την ίδια απόσταση και όταν τους αντιλαμβάνεται το ανθρώπινο αυτί.

### *4.2.6 Διάγραμμα MFCC Χαρακτηριστικών*

Τα διαγράμματα των *MFCC* χαρακτηριστικών μας παρέχουν μία πληθώρα από ουσιώδης πληροφορίες για το εκάστοτε ηχητικό σήμα.

Είναι μία πιο συμπιεσμένη αναπαράσταση συγκριτικά με τα διαγράμματα *Mel-Spectrogram,* καθώς χρησιμοποιούνται μόνο 20 ή 13 συντελεστές (*στην περίπτωση μας χρησιμοποιούνται 20*). Ακόμη, για την εξαγωγή των MFCC χαρακτηριστικών γίνεται ο υπολογισμός του *DCT* στο *Mel-Spectrogram*.

Δείτε παρακάτω μία ενδεικτική εικόνα από τα τρία τελευταία διαγράμματα:




*Εικόνα 4: Διαγράμματα Spectrogram - Mel Spectrogram - MFCC*

# 5. Συμπεράσματα

Μετά την εκτέλεση του προγράμματος καταλήγουμε στα εξής συμπεράσματα:

1. Παρατηρούμε ότι τα ποσοστά αναγνώρισης των ψηφίων μετά την εκτέλεση των μετρικών είναι 78.57%, κάτι το οποίο είναι λογικό αφού είναι πολύ δύσκολο να επιτευχθεί πλήρη ταύτιση των προβλέψεων με τις πραγματικές τιμές.
2. Με τη βοήθεια του Confusion matrix μπορούμε να εντοπίσουμε το πλήθος των φορών που ένα ψηφίο δεν αναγνωρίστηκε επιτυχώς, αλλά και την αντιστοίχιση με την εσφαλμένη πρόβλεψη.

## 6. Απαιτήσεις συστήματος και εκτέλεση

Η παραπάνω άσκηση υλοποιήθηκε στην γλώσσα προγραμματισμού ***Python, v3.7.9***. Για να μπορεί να γίνει η εκτέλεση του προγράμματος είναι απαραίτητο να είναι εγκατεστημένες οι βιβλιοθήκες:

- **librosa**
- **numpy**
- **sklearn**
- **matplotlib**
- **os**
- **time**

Εάν στον υπολογιστή σας δεν έχει γίνει εγκατάσταση των βιβλιοθηκών **librosa**, **numpy** προηγουμένως, τρέξτε τις εντολές ***«pip install librosa»***, **«***pip install numpy***»** στο command line.

Σε περίπτωση που η βιβλιοθήκη **sklearn** δεν είναι εγκατεστημένη, τότε τρέχουμε την εντολή **«***pip install scikit-sklearn***»** στο command line.

Επίσης, ο χρήστης θα πρέπει να έχει προβλέψει τα αρχεία του συνόλου εκπαίδευσης και το αρχείο εισόδου να είναι στον φάκελο **training** και **testing**, αντίστοιχα.

Οι συναρτήσεις που υλοποιήθηκαν βρίσκονται στο αρχείο **functions.py**. Το αρχείο που καλεί τις συγκεκριμένες συναρτήσεις και είναι υπεύθυνο για την εκτέλεση του προγράμματος είναι το αρχείο **main.py**.

Για να εκτελέσουμε τον κώδικα ακολουθούμε τα παρακάτω βήματα:

1. Ανοίγουμε τη γραμμή εντολών και μεταβαίνουμε στον αντίστοιχο φάκελο**: \Επεξεργασία Συστημάτων Φωνής και Ήχου\Θέμα 1**. (***Εικόνα 5***)
2. Πληκτρολογούμε το όνομα του αρχείου **main.py**, ώστε να ανοίξει το εκτελέσιμο. (***Εικόνα 6***)

Το αρχείο εκτελείται και εμφανίζεται το αποτέλεσμα της αναγνώρισης των ψηφίων. (***Εικόνα 7***)

Βγάζοντας τα σχόλια για την κλήση των συναρτήσεων ***plots*** και ***accuracy*** εμφανίζονται τα αποτελέσματα, όπως έχει περιγραφεί στην Ενότητα 4.

Αξιοσημείωτο είναι, ότι στο αρχείο **requirements.txt** υπάρχουν οι απαιτήσεις του προγράμματος. Ο χρήστης μπορεί να εκτελέσεις την παρακάτω εντολή για να εγκαταστήσει τις βιβλιοθήκες που απαιτούνται. Για την εν λόγω λειτουργία, και αφού προηγουμένως έχουμε μεταβεί στον φάκελο **\Επεξεργασία Συστημάτων Φωνής και Ήχου\Θέμα 1** (***Εικόνα 5***), εκτελούμε στο command line την εντολή «*pip install -r requirements.txt*».

```
Microsoft Windows [Version 10.0.19042.1083]
(c) Microsoft Corporation. Με επιφύλαξη κάθε νόμιμου δικαιώματος.

C:\Users\panap\OneDrive\Υπολογιστής\Εργασία Συστημάτων Φωνής και Ήχου>cd Θέμα 1
```

*Εικόνα 5: Μετάβαση στον φάκελο του εκτελέσιμου*

```
Microsoft Windows [Version 10.0.19042.1083]
(c) Microsoft Corporation. Με επιφύλαξη κάθε νόμιμου δικαιώματος.

C:\Users\panap\OneDrive\Υπολογιστής\Εργασία Συστημάτων Φωνής και Ήχου>cd Θέμα 1

C:\Users\panap\OneDrive\Υπολογιστής\Εργασία Συστημάτων Φωνής και Ήχου\Θέμα 1>main.py
```

*Εικόνα 6: Εισαγωγή του ονόματος αρχείου προς εκτέλεση*

```
Microsoft Windows [Version 10.0.19042.1083]
(c) Microsoft Corporation. Με επιφύλαξη κάθε νόμιμου δικαιώματος.

C:\Users\panap\OneDrive\Υπολογιστής\Εργασία Συστημάτων Φωνής και Ήχου>cd Θέμα 1

C:\Users\panap\OneDrive\Υπολογιστής\Εργασία Συστημάτων Φωνής και Ήχου\Θέμα 1>main.py
[Process]: Preprocessing Started
[Process]: Preprocessing Completed
[Process]: Dataset Training Started
[Process]: Dataset Training Completed
[Process]: RMSE Calculated
[Process]: ZCR Calculated
[Process]: Background vs Foreground Classification Started
[Process]: Background vs Foreground Classification Completed
[Process]: Digits Recognition Started
[Process]: Digits Recognition Completed
[Result]: ['3' '8' '6' '4' '0']
```

*Εικόνα 7: Αποτελέσματα εκτέλεσης*

# 7. Βιβλιογραφία

Σε αυτή την ενότητα θα αναφερθούν οι βιβλιογραφικές πηγές της εφαρμογής μας:

1. Visual Studio Code Documentation
   *(τελευταία προσπέλαση 03/07/2021)*
2. Audacity Documentation
   *(τελευταία προσπέλαση 11/07/2021)*
3. Python v.3.7.9 Documentation
   *(τελευταία προσπέλαση 14/07/2021)*
4. NumPy v1.21 Documentation
   *(τελευταία προσπέλαση 08/07/2021)*
5. Librosa v0.8.1 Documentation
   *(τελευταία προσπέλαση 14/07/2021)*
6. Scikit-Learn v0.24.2
   *(τελευταία προσπέλαση 12/07/2021)*
7. Matplotlib v3.4.2
   *(τελευταία προσπέλαση 12/07/2021)*
8. FIR Band System Theory & Design Examples
   *(τελευταία προσπέλαση 02/07/2021)*
9. Digital Signal Processing
   *(τελευταία προσπέλαση 02/07/2021)*
10. Digital Audio Basics: Audio Sample Rate and Bit Depth
    *(τελευταία προσπέλαση 02/07/2021)*
11. Sampling Rates, Sample Depths, and Bit Rates: Basic Audio Concepts
    *(τελευταία προσπέλαση 03/07/2021)*
12. MFCC Features
    *(τελευταία προσπέλαση 11/07/2021)*
13. Semantic Scholar - A free, AI-powered research tool for scientific literature
    *(τελευταία προσπέλαση 05/07/2021)*
14. A tutorial on signal energy and its applications – Rodrigo Capobianco Guido (via Semantic Scholar)
    *(τελευταία προσπέλαση 05/07/2021)*
15. Detection and Recognition Threshold of Sound Sources in Noise - Tjeerd Andringa, Carina Pal (via Semantic Scholar)
    *(τελευταία προσπέλαση 06/07/2021)*

16. BF-Classifier: Background/Foreground Classification and Segmentation of Soundscape Recordings - Miles Thorogood, Jianyu Fan, Philippe Pasquier (via Semantic Scholar)
*(τελευταία προσπέλαση 10/07/2021)*
17. Short-time energy, magnitude, zero crossing rate and autocorrelation measurement for discriminating voiced and unvoiced segments of speech signals - Madiha Jalil, F. A. Butt, Ahmed Malik (via Semantic Scholar)
*(τελευταία προσπέλαση 08/07/2021)*
18. Separation of Voiced and Unvoiced using Zero crossing rate and Energy of the Speech Signal - Bachu R.G., Kopparthi S., Adapa B., Barkana B.D. (via Semantic Scholar)
*(τελευταία προσπέλαση 09/07/2021)*
19. Automatic Silence/Unvoiced/Voiced Classification of Speech Using a Modified Teager Energy Feature - Alexandru Caruntu, Gavril Toderean, Alina Nica
*(τελευταία προσπέλαση 12/07/2021)*
20. Separation of Voiced and Unvoiced using Zero crossing rate and Energy of the Speech Signal - Bachu R.G., Kopparthi S., Adapa B., Barkana B.D
*(τελευταία προσπέλαση 09/07/2021)*
21. Cylinder misfire detection using engine sound quality metrics and random forest classifier - Sneha Singh, Sagar Potala and Amiya Ranjan Mohanty
*(τελευταία προσπέλαση 14/07/2021)*
22. Comparison Of Five Classifiers For Classification Of Syllables Sound Using Time-Frequency Features - Domy Kristomo, Risanuri Hidayat, Indah Soesanti
*(τελευταία προσπέλαση 13/07/2021)*
23. Understanding Confusion Matrix
*(τελευταία προσπέλαση 12/07/2021)*
24. Mel-spectrogram augmentation for sequence-to-sequence voice conversion - Yeongtae Hwang , Hyemin Cho, Hongsun Yang, Dong-Ok Won, Insoo Oh and Seong-Whan Lee
*(τελευταία προσπέλαση 10/07/2021)*

25. Spoken Digits Recognition using Weighted MFCC and Improved Features for Dynamic Time Warping - V.Chapaneri, Santosh
*(τελευταία προσπέλαση 11/07/2021)*